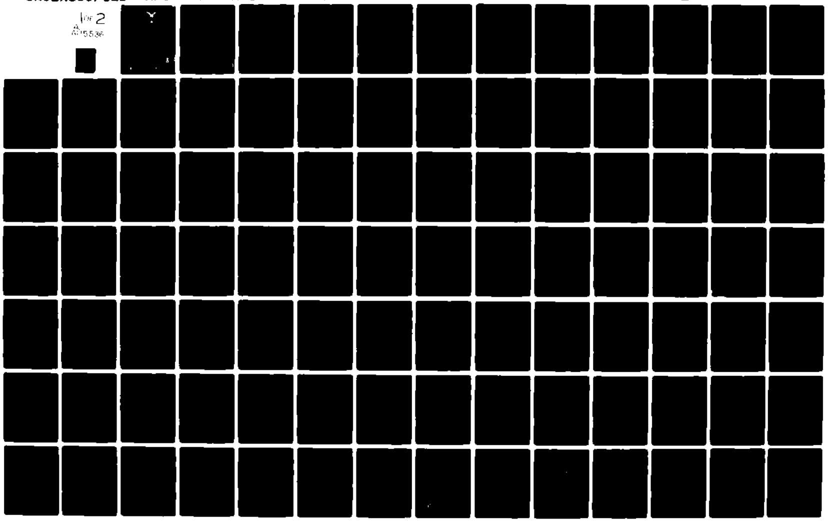
AD-A115 536   AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOO--ETC F/G 15/7
A COMPUTERIZED ALGORITHM FOR SOLVING MULTI-STAGE SIMULTANEOUS G--ETC(U)
DEC 81 M A FATEEN
UNCLASSIFIED AFIT/GOR/MA/81D-5 NL

1 of 2
A
115536



UNITED STATES AIR FORCE

AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GOR/MA/81D-5

1

A COMPUTERIZED ALGORITHM FOR SOLVING MULTI-STAGE SIMULTANEOUS GAMES

THESIS

Mohamed Abdelrahman Fateen
Lieutenant Colonel, Egyptian Air Force

AFIT/GOR/MA/81D-5

AFIT/GOR/MA/81D-5

A COMPUTERIZED ALGORITHM FOR SOLVING MULTI-STAGE SIMULTANEOUS GAMES

THESIS

Presented to the Faculty of the School of Engineering

of the Air Force Institute of Technology

in Partial Fulfillment of the

Requirements for the Degree of

Master of Science

by

Mohamed Abdelrahman Fateen
Lieutenant Colonel, Egyptian Air Force

Graduate Operations Research

December 1981

Approved for public release; distribution unlimited

Preface

Multi-stage games have long been of great interest to me. When, as an officer in the Egyptian Air Force, I joined the Operations Research Center in Cairo, Egypt, I was involved in computerization of the Air War Board Game from Simulations Publication, Inc. (SPI). The question of what is the "optimal" strategies in playing those games had no answer for me. While at AFIT I learned that multi-stage game theory may be used to approximate the optimal solution. Hence, I decided to develop a multi-stage game algorithm. The algorithm developed in this thesis finds the game value and one of the optimal strategies for each stage of the game using both single stage game theory and dynamic programming.

I wish to thank Lieutenant Colonel James Bexfield, my faculty advisor, for his assistance and sound advice throughout this effort. In addition, thanks are extended to my reader, Lieutenant Colonel Charles McNichols, for his time and encouragement; and to my typist, Phyllis Reynolds, for her tireless efforts to produce a quality product.

Finally, I wish to thank my wife, Sohair, whose usually gentle prodding ensured the successful completion of this research.

— Mohamed A. Fateen

Contents

Page

List of Figures

List of Tables

Abstract

The algorithm developed in this study finds the solution to multi-stage simultaneous games. A dynamic programming approach is used to solve the multi-stage game. The main idea of the solution is to build at each stage a matrix whose values are the payoffs obtained by playing each of the given strategies at this stage and the optimal strategies for the remaining stages. This payoff matrix is then solved using a linear programming algorithm.

A tactical air war problem was selected to illustrate the usefulness of the algorithm. An interactive GAME was developed for training and assisting the field commander. Finally, it was shown how the multi-stage algorithm can be used to compare two weapon systems by letting each side play its optimal strategy.

A COMPUTERIZED ALGORITHM FOR SOLVING MULTI-STAGE SIMULTANEOUS GAMES

I. Introduction

> The theory of games of strategy may be described as a mathematical theory of decision-making by participants in a competitive environment. In a typical problem to which the theory is applicable, each participant can bring some influence to bear upon the outcome of a certain event; no single participant by himself nor chance alone can determine the outcome completely. The theory is then concerned with the problem of choosing an optimal course of action which takes into account the possible actions of the participants and the chance events [Ref 3:1].

Examples of games of strategy include poker, chess, and military battles. Each of these games allows the players to make use of their ingenuity in order to influence the outcome.

The problem we are dealing with is a particular form of game theory--that for a multi-stage or an "N-stage game." The name reflects the fact that there are a series of N decision points, or stages, in the conduct of a tactical campaign. The proper allocation of forces at each stage depends upon the cumulative outcome of the preceding stages and the length of time left in the campaign. The solution to a multi-stage game indicates the strategies each side should play so as to optimize their respective payoffs.

Others have attempted to solve multi-stage games. We review three optimization models OPTSA (Ref 2), Lulejian model (Ref 10), and DYGAM (Ref 6:A-1 to A-12). OPTSA's insistence on exact optimality causes long running times and limits the number of stages in the game. Lulejian models use successive sweep techniques to find an optimal solution. DYGAM uses the dynamic programming approach to get the upper and lower bounds and enforceable strategies, corresponding closely to the minimax and maximin strategies for the game at a stage (Refs 6; 16).

There are two research objectives:

1. Develop an algorithm and computer code to solve the simultaneous multi-stage game. This algorithm will incorporate the best features of the available existing models, and use new concepts that either improve the efficiency of the model or increase its realism.

2. Illustrate how game theory can be used in national defense. This includes exploring the usefulness of game theory as a training device (Chapter IX) and as an aid to procurement decision making (Chapter X).

The algorithm developed is based on the concept of dynamic programming (see Appendix B). In this sense its basic structure parallels that of the DYGAM model. A more detailed comparison is not possible since only very general documentation was available for the DYGAM model.

The remainder of this thesis will pursue the objectives previously outlined. The second chapter will present a review of single-stage game theory. This provides the reader with a basic understanding of the terminology of game theory and how linear programming can be used to solve the single-stage game. The third chapter describes the general multi-stage game problem. The fourth chapter is a review of the literature for the three models mentioned above. The fifth chapter contains the general algorithm. The author selected the tactical air war problem to illustrate the usefulness of the algorithm. The sixth chapter describes the tactical air war problem. Chapter VII describes the linear model of the tactical air war together with the solution algorithm needed to solve it. Chapter VIII indicates how the algorithm developed in Chapter VII was verified and validated. Chapter IX describes an application of the linear model which may be useful in training Air Force decision makers. Chapter X shows how the model could potentially be used to give insight into an aircraft procurement decision. Chapter XI describes a modification to the linear model. Finally, Chapter XII provides conclusions and recommendations for further research.

II. Game Theory Overview

Introduction

This chapter is designed to give the reader who has a general operations research background a basic understanding of game theory. The terminology and concepts discussed in this chapter will be used in the development of the algorithm used to solve the multistage game. First we define what we mean by a payoff, then we discuss what is meant by strategies, and a two-person zero-sum game. Next the concepts of an optimal solution for a two-person zero-sum game is discussed. A graphical solution technique is explained. Finally we show how the technique of linear programming can be used to solve two-person zero-sum games.

Game Theory

Game theory (Ref 12:339-352) deals with decisions under uncertainty involving two or more intelligent opponents in which each opponent aspires to optimize his own decision but at the expense of the other opponents. Typical examples include launching advertisement campaigns for competing products and planning war tactics for opposing armies.

In game theory, an opponent is referred to as a player. Each player has a number of choices, finite or infinite in number, called strategies. The outcomes or payoffs of a game depend on the strategies selected by each of the players. A game with two players, where the gain of one player equals the loss to the other, is known as a two-person zero-sum game. In such a game it suffices to express the outcomes in terms of the payoff to one player. A matrix is usually used to summarize the payoffs to the player whose strategies are given by the rows of the matrix while the columns represent the corresponding strategies for his opponent.

Optimal Solution of Two-Person Zero-Sum Games

Games represent the ultimate case of lack of information in which intelligent opponents are working in a conflicting environment (Ref 12:340). The result is that a very conservative criterion is often proposed for solving two-person zero-sum games. This is the minimax-maximin criterion.

To accommodate the fact that each opponent is working against the other's interest, the minimax criterion selects strategies for each player which will result in the best of the worst possible outcomes. Since the game matrix is usually expressed in terms of the payoff to player A (whose strategies are represented by the rows), the criterion calls for A to select the strategy which maximizes his minimum gain, the minimum being taken over all the strategies of player B. By the same reasoning, player B selects his strategy which minimizes his maximum losses. The maximum is taken over all A's strategies.

As an example, consider a two-person zero-sum game with each player able to select one of four options (strategies) at each play of the game. The entries in the payoff matrix, Table II-1, represent player A's gain.

TABLE II-1

TWO-PERSON ZERO-SUM EXAMPLE PAYOFF TABLE

		PLAYER B					
		1	2	3	4	Row Minimum	
	1	9	3	8	4	3	
PLAYER A	2	6	4	7	10	④	Maximin
	3	8	3	5	-7	-7	
	4	2	3	8	6	2	
Column Maximum:		9	④ Mini-max	8	10		

When player A plays his first strategy, he may gain 9, 3, 8 or 4 depending on player B's selected strategy. He can guarantee, however, a gain of at least min{9,3,8,4}=3 regardless of B's selected strategy. Similarly, if A plays his second strategy, he guarantees a payoff of at least min{6,4,7,10}=4; if he plays his third strategy, he guarantees a payoff of at least min{8,3,5,-7}=-7; and if he plays his fourth strategy, he guarantees a payoff of at least min{2,3,8,6}=2. Thus, the minimum values in each row represents the minimum gain guaranteed A if he plays the pure strategy associated with that row. Now player A, by selecting his second strategy, assures himself of a payoff of at least 4. He is maximizing his minimum gain. Player A's selection is called the maximin strategy, and his corresponding gain is called the maximin (or lower) value of the game.

Player B, on the other hand, wants to minimize his losses. He realizes that, if he plays his first pure strategy (first column) he can lose no more than max{9,6,8,2}=9, regardless of A's selections. The corresponding results are thus indicated in the above matrix by "column maximum." Player B will then select the strategy that minimizes his maximum loss. This is given by the second strategy and his corresponding loss is given by min{9,4,8,10}=4. Player B's selection is called the minimax strategy and his corresponding loss is called the minimax (or upper) value of the game.

From the conditions governing the minimax criterion, the minimax (upper) value is greater than or equal to the maximin (lower) value. In the case where the equality holds (i.e., minimax value = minimin value), the corresponding pure strategies are called "optimal" strategies and the game is said to have a saddle point. The value of the game is equal to the common value of the maximin and minimax values.

The fact that this game possesses a saddle point (Ref 7:305) was crucial in determining how it should be played. If the game possesses a saddle point then neither player can take advantage of the opponent's strategy to improve his own position. In particular, when player A predicts or learns that player B is using his second strategy, player A would only decrease his gain if he were to change from his original plan of using his second strategy. Similarly, player B would only worsen his position if he were to change his plan. These maximin and minimax strategies are said to be stable strategies and an optimal solution to the game has been found.

Mixed Strategies

The above section shows that the existence of a saddle point immediately yields the optimal pure strategies for the game. However, some games do not have saddle points. For example, consider the two-person zero-sum game in Table II-2. The minimax value (8) is greater than the maximin value (2). Hence the game does not have a saddle point and the pure maximin-minimax strategies are not optimal. Now knowledge of the opponent's strategy can be used to improve a player's payoff. In this case the game is said to be unstable.

The failure of the minimax-maximin (pure) strategies, in general, to give an optimal solution to the game has led to the idea of using mixed strategies (12:342). Under this concept each player selects the strategy to play at random according to a predetermined set of probabilities.

TABLE II-2

PAYOFF TABLE WITHOUT SADDLE POINT

		PLAYER B 1	2	3	4	Row Minimum
PLAYER A	1	9	7	-3	8	-3
	2	8	9	5	4	$\textcircled{4}$ Maximin
	3	1	2	9	7	1
	4	10	8	2	5	2
Column Maximum:		10	9	9	$\textcircled{8}$ Mini-max	

Let $x_1, x_2, \ldots, x_m$ and $y_1, y_2, \ldots, y_n$ be the row and column probabilities by which A and B, respectively, select their strategies. Note that the x_i and y_i selected must satisfy

$$\sum_{i=1}^{m} x_i = \sum_{j=1}^{n} y_i = 1 \qquad (2\text{-}1)$$

$$x_i, y_j \geq 0 \quad \text{for all } i \text{ and } j.$$

Suppose player A has m pure strategies available to him and player B may select any one of n strategies. Let a_{ij} represent the payoff to A if A selects strategy i and B selects strategy j. Then Table II-3 contains the full payoff matrix.

In general, the value of the above game must satisfy the inequality

MAXIMIN (LOWER) VALUE $\leq$ VALUE OF THE GAME
$\leq$ MINIMAX (UPPER) VALUE

TABLE II-3

GENERAL PAYOFF MATRIX

		PLAYER B			
		1	2	...	n
PLAYER A	1	a_{11}	a_{12}	...	a_{1n}
	2	a_{21}	a_{22}	...	a_{2n}
	⋮	⋮	⋮		⋮
	n	a_{n1}	a_{m2}	...	a_{mn}

The actual solution of the mixed strategy problem is based on the minimax criterion given before. The only difference is that A selects the $x_1, x_2, \ldots, x_m$ which maximize the smallest expected payoff in a column, while B selects the $y_1, y_2, \ldots, y_n$ which minimize the largest expected payoff in a row. Mathematically, the minimax criterion for a mixed strategy case is given as follows: player A selects the $x_1, x_2, \ldots, x_n$ that

$$\underline{V} = \max \{\min(\sum_{i=1}^{m} a_{i1}x_i, \sum_{i=1}^{m} a_{i2}x_i, \ldots, \sum_{i=1}^{m} a_{in}x_i)\} \qquad (2\text{-}2)$$

and player B selects the $y_1, y_2, \ldots, y_m$ that

$$\overline{V} = \min \{\max(\sum_{j=1}^{n} a_{1j}y_j, \sum_{j=1}^{n} a_{2j}y_j, \ldots, \sum_{j=1}^{n} a_{mj}y_j)\} \qquad (2\text{-}3)$$

These values are referred to as the maximum and the minimax expected payoffs, respectively.

As in the pure strategies case, the minimax expected payoff is always greater than or equal to maximin expected payoff ($\overline{V} \geq \underline{V}$). When $x_1, x_2, \ldots, x_n$ and $y_1, y_2 \ldots, y_m$ correspond to the optimal solution, the equality holds and the resulting expected payoffs become equal to the value of the game. This result follows from the minimax theorem (Ref 12:343): suppose $x_1^*, x_2^*, \ldots x_n^*$ and $y_1^*, y_2^*, \ldots, y_m^*$ are the optimal sets of probabilities for both players. Then, the optimal expected value of the game is

$$\overline{V} = v^* = \sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij} x_i^* y_j^* = \underline{V} \; . \qquad (2\text{-}4)$$

There are several methods of solving two-person zero-sum games for the optimal values of $x_1, x_2, \ldots, x_n$ and $y_1, y_2, \ldots, y_m$. The next section presents the graphical method for solving a game where one of the two sides has exactly two strategies available to him (i.e., a (2xn) or (mx2) game). A general linear programming method for solving any mxn game is presented in the last section of this chapter.

Graphical Solution of (2xn) and (mx2) Games

Graphical solutions are only applicable to games in which at least one of the players has exactly two strategies. Consider the (2xn) game in Table II-4. Assume that the game does not have a saddle point. Since A has

TABLE II-4

(2xn) TWO-PERSON ZERO-SUM GAME

$$\begin{array}{ll|cccc}
 & & & \text{PLAYER B} & & \\
 & & y_1 & y_2 & \cdots & y_n \\
\hline
\text{PLAYER A} & x_1 & a_{11} & a_{12} & \cdots & a_{1n} \\
 & x_2=1-x_1 & a_{21} & a_{22} & \cdots & a_{2n}
\end{array}$$

two strategies, it follows that $x_2=1-x_1$; $x_1\geq 0$, $x_2\geq 0$. His expected payoffs corresponding to the pure strategies of B are given in Table II-5.

TABLE II-5

A's EXPECTED PAYOFFS

B's Pure Strategy	A's Expected Payoff
1	$(a_{11}-a_{21})x_1+a_{21}$
2	$(a_{12}-a_{22})x_1+a_{22}$
⋮	⋮
n	$(a_{1n}-a_{2n})x_1+a_{2n}$

Note that A's average payoff varies linearly with x_1.

According to the minimax criterion for mixed strategy games, player A should select the value of x_1 which maximizes his minimum expected payoffs. This may be done by plotting the above straight lines as a function of x_1. A typical example is illustrated in Figure II-1. Each line is numbered according to the corresponding pure

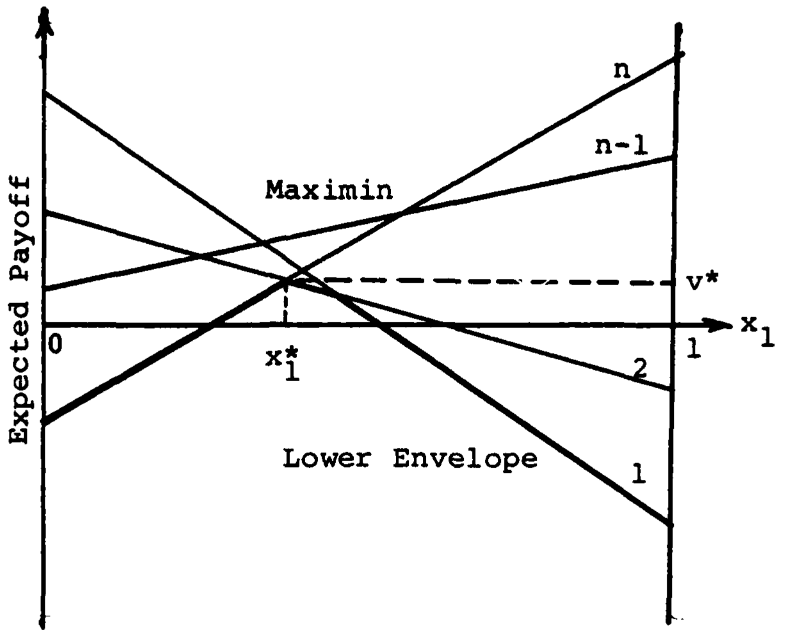


Fig. II-1. Graphical Solution to (2xn) Game (Player A)

strategy for B. The lower envelope of these lines (indicated by heavy line segments) gives the minimum expected payoff as a function of x_1. The highest point on this lower envelope (indicated by a dot) gives the maximin expected payoff and hence the optimum value of $x_1 (=x_1^*)$.

The optimal y_j for B can be obtained by observing the definition of the expected value of the game. This is given for the above (2xn) game by

$$v^* = y_1^*\{(a_{11}-a_{21})x_1^*+a_{21}\}+y_2^*\{(a_{12}-a_{22})x_1^*+a_{22}\} + \dots +y_n^*\{(a_{1n}-a_{2n})x_1^*+a_{2n}\} \qquad (2\text{-}5)$$

All the lines $(a_{1j}-a_{2j})x_1^*+a_{2j}$ that do not pass through the maximin point must have their corresponding $y_j^*=0$. This follows since at x_1^* any of these lines (for example line 1 in Figure II-1) will yield an expected payoff for A greater than the maximin expected payoff, a result which violates the minimax theorem.

Because the maximin point is determined by the intersection of two straight lines, it follows that, except for the y_j corresponding to these two straight lines, all other y_j may be taken equal to zero. If, however, there are more than two lines passing through the maximin point, any two lines having opposite slopes may be selected to determine the optimal values of y_j. Each resulting solution is an alternative solution. Consequently, any weighted average of these solutions will also yield a new optimal solution.

The above discussion reveals that any (2xn) game is basically equivalent to a (2x2) game. Let y_{j1} and y_{j2} be the probabilities corresponding to the "active" strategies of B. Let all other $y_i=0$. Then B's expected payoffs corresponding to A's pure strategies are given in Table II-6.

TABLE II-6

B's EXPECTED PAYOFFS

A's Pure Strategy	B's Expected Payoff
1	$(a_{1j1}-a_{1j2})y_{j1}+a_{1j2}$
2	$(a_{2j1}-a_{2j2})y_{j1}+a_{2j2}$

These two lines are then plotted as a function of y_{j1} (see Figure II-2). Since B wishes to minimize his maximum expected payoff, the minimum point of the upper envelope of these two lines identifies y^*_{j1}.

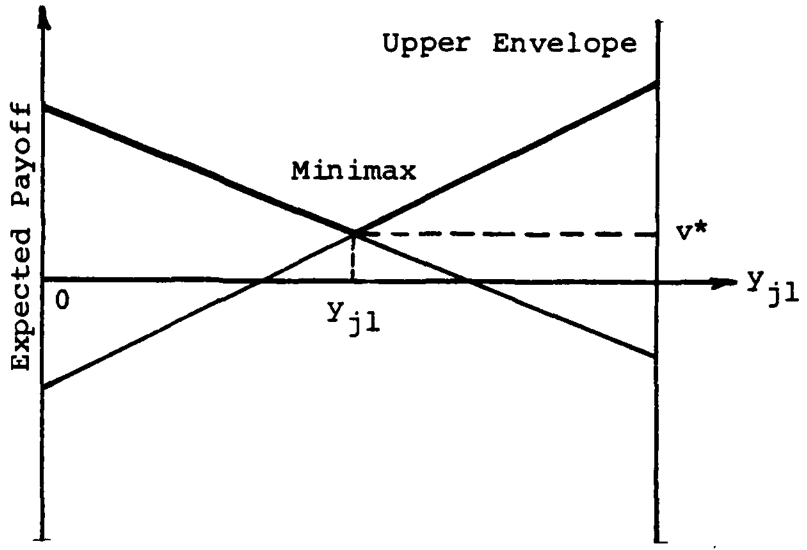


Fig. II-2. Graphical Solution to (2xn) Game (Player B)

Solution of (mxn) Games by Linear Programming

Game theory bears a strong relationship to linear programming since every finite two-person zero-sum game can be expressed as a linear program and, conversely, every linear program can be represented as a game (12:346). This section will explain how linear programming can be used to find the solution to two-person zero-sum games. It is especially useful for games with large payoff matrices.

Recall (equation 2-2) that A's optimum mixed strategies satisfy

$$\max\{\min(\sum_{i=1}^{m} a_{i1}x_i,\ \sum_{i=1}^{m} a_{i2}x_i,\ \ldots,\ \sum_{i=1}^{m} a_{in}x_i)\} \qquad (2\text{-}2)$$

subject to the constraints

$x_1+x_2+\ldots+x_m=1$

$x_i \geq 0, \quad i=1,2,\ldots,m$

Let

$$v = \min(\sum_{i=1}^{m} a_{i1}x_1,\ \sum_{i=1}^{m} a_{i2}x_i,\ldots,\ \sum_{i=1}^{m} a_{in}x_i) \qquad (2\text{-}6)$$

Then the problem becomes

$$\text{maximize } z_0=v \qquad (2\text{-}7)$$

subject to

$\sum_{i=1}^{m} a_{ij}x_i \geq v, \quad j=1,2,\ldots,n$

$\sum_{i=1}^{m} x_i=1$

$x_i \geq 0$, for all i.

Note that v represents the value of the game.

The above linear programming formulation can be simplified by dividing all (n+1) constraints by v. This division is correct as long as v>0. If v<0, the direction of the inequality constraints must be reversed. If v=0, the division is illegitimate. This point presents no special problem since a positive constant Z can be added to all the entries of the matrix, thus guaranteeing that the value of the game for the "modified" matrix will be greater than zero. After the optimal solution is obtained, the true value of the game is determined by subtracting Z.

Thus, assuming that $v>0$, the constraints of the linear programming become

$$\begin{aligned}
a_{11}\frac{x_1}{v} + a_{21}\frac{x_2}{v} + \dots + a_{m1}\frac{x_m}{v} &\geq 1 \\
a_{12}\frac{x_1}{v} + a_{22}\frac{x_2}{v} + \dots + a_{m2}\frac{x_m}{v} &\geq 1 \\
\vdots \qquad\qquad \vdots \qquad\qquad & \vdots \\
a_{1n}\frac{x_1}{v} + a_{2n}\frac{x_2}{v} + \dots + a_{mn}\frac{x_m}{v} &\geq 1
\end{aligned} \tag{2-8}$$

$$\frac{x_1}{v} + \frac{x_2}{v} + \dots + \frac{x_m}{v} = \frac{1}{v}$$

Let $x_i = x_i/v$, $i=1,2,\dots,m$. Since

$$\max v = \min \frac{1}{v} = \min\{x_1+x_2+\dots+x_m\}$$

the problem becomes

$$\text{minimize } x_0 = x_1+x_2+\dots+x_m$$

subject to

$$\begin{aligned}
a_{11}x_1 + a_{21}x_2 + \dots + a_{m1}x_m &\geq 1 \\
a_{12}x_1 + a_{22}x_2 + \dots + a_{m2}x_m &\geq 1 \\
&\vdots \\
a_{1n}x_1 + a_{2n}x_n + \dots + a_{mn}x_m &\geq 1
\end{aligned} \qquad (2\text{-}9)$$

$$x_1, x_2, \dots x_m \geq 0.$$

Player B's problem is given by

$$\min_{Y_j} \{\max(\sum_{j=1}^{n} a_{ij}y_j, \sum_{j=1}^{n} a_{2j}y_j, \dots, \sum_{j=1}^{n} a_{mj}y_m)\} \qquad (2\text{-}10)$$

subject to

$$y_1 + y_2 + \dots + y_n = 1$$

$$y_j \geq 0, \quad j=1,2,\dots,n$$

The linear programming formulation is:

$$\text{Maximize } Y_0 = Y_1 + Y_2 + \dots + Y_n$$

subject to

$$\begin{aligned}
a_{11}Y_1 + a_{12}Y_2 + \dots a_{1n}Y_n &\leq 1 \\
a_{21}Y_1 + a_{22}Y_2 + \dots a_{2n}Y_n &\leq 1 \\
&\vdots \\
a_{m1}Y_1 + a_{m2}Y_2 + \dots a_{mn}Y_n &\leq 1
\end{aligned} \qquad (2\text{-}11)$$

$$Y_1, Y_2, \dots Y_n \geq 0$$

where $y_0 = \frac{1}{v}$, $Y_j = \frac{y_j}{v}$, $j=1,2,\ldots,n$

Notice that B's problem is actually the dual of A's problem. Thus the optimal solution of one problem automatically yields the optimal solution to the other. Player B's problem can be solved by the regular simplex method, while player A's problem is solved by the dual simplex method.

III. A Multi-Stage Simultaneous Game

In the preceding chapter we provided an overview of a single-stage game theory. The players of a single-stage game are only concerned with maximizing or minimizing the single-stage game value. Although these single-stage games can be played repetitively, each play is independent of all previous plays. The multi-stage gaming problem is more complex. The outcome of one play of the multi-stage game may influence the resources each player may use in a subsequent play. There are two types of multi-stage games--sequential and simultaneous. In the simultaneous game, both players redeploy during each stage (neither side knows the other's deployment decision when he makes his deployment decision); while in a sequential game, one side deploys first and then the other side redeploys with full knowledge of the other's decision. Sequential games are not considered in this thesis. Next, we formally define a multi-stage simultaneous game.

A multi-stage simultaneous game consists of the following:

1. Two players engage in a sequence of two-person, zero-sum games. One game corresponds to a stage in the sequence.

2. Both players have complete information about all preceding stages.

3. Both players redeploy during each stage.

4. The payoffs on each stage are determined not only by the strategies chosen by both players at this stage, but also by the values of a set of resources or other variables which serve to characterize the state of the game at the beginning of the stage. A vector $(x_1, x_2, \ldots x_N)$, called the state variable, is often used to represent the state of the game at the start of each stage.

5. The strategies chosen by the players together with the current state determine the value of the state at the beginning of the next stage. The state transition function is used to determine the state vector for the next stage from the state vector for the last stage and the players' game strategies.

6. The total payoff for the game is the sum of the incremental payoffs on each of the stages.

7. The objective of the game is to maximize (in the case of the Blue player) or minimize (in the case of the Red player) the total payoff for the game.

Figure III-1 shows the two-stage game. The result of the first stage depends on the resources available to both sides and the strategies they choose. The first stage game will result in a certain payoff for Blue (which is the negative of the Red payoff--recall the zero-sum

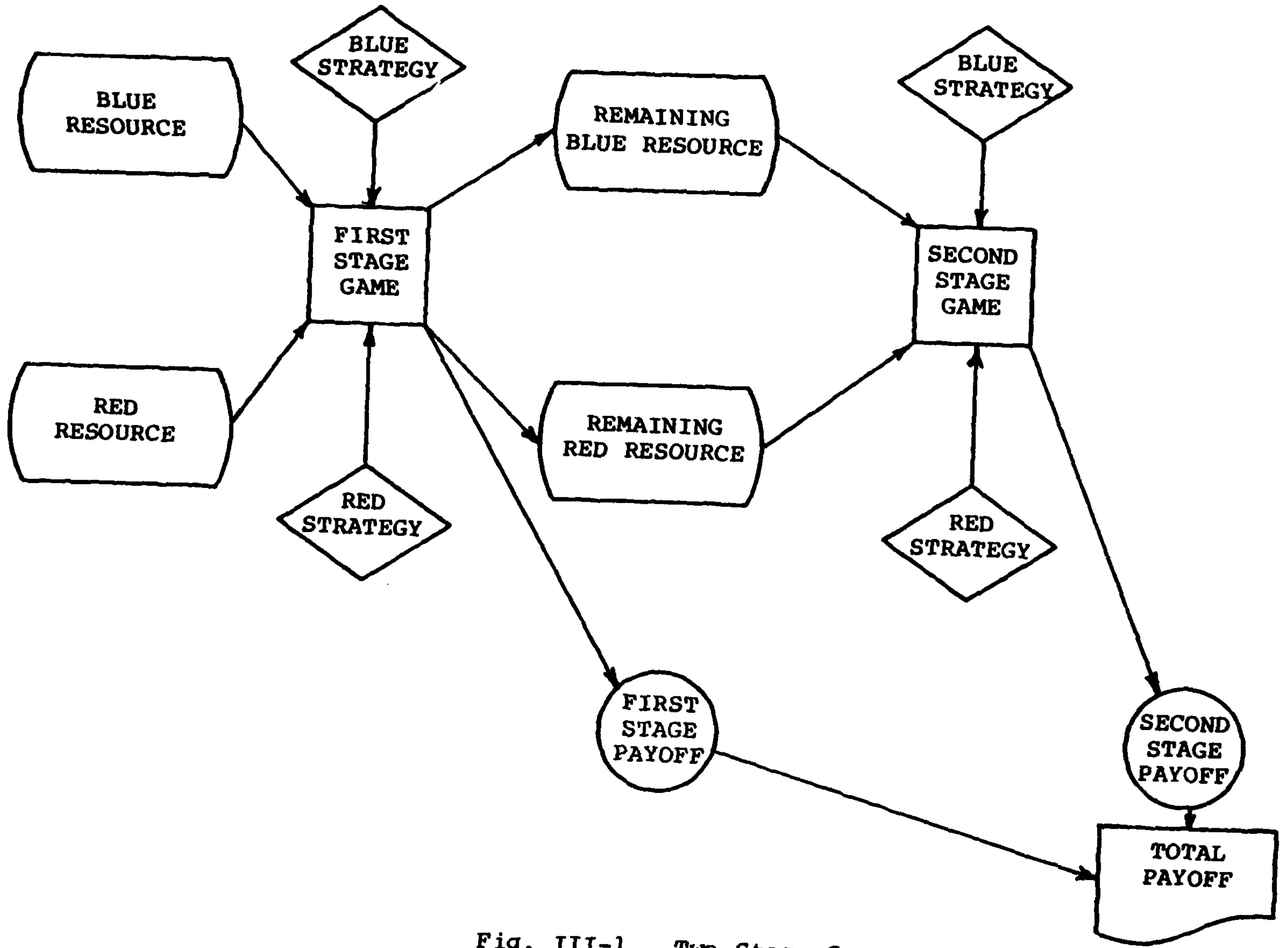


Fig. III-1. Two-Stage Game

concept) and the remaining resources available to both sides. The second game payoff depends on the resources remaining from the previous play (recall we assume no replenishment) and the strategies both players use.

IV. Literature Survey

In this chapter we survey three multi-stage optimization models. These are the OPTSA (Ref 2), Lulejian (Ref 10), and DYGAM (Ref 6:A-1 to A-12) models. These models deal with two-person, zero-sum simultaneous multi-stage games. OPTSA obtains the exact optimal solution but at the expense of long computer times and a limited number of stages (three stages). Lulejian has a weakness in the optimization methodology and allows a limited number of decision vectors (Ref 6:11). DYGAM uses the technique of dynamic programming.

In addition, the author learned a great deal from the book Games of Strategy by Melvin Dresher (Ref 3). A discussion of a solution developed by Dresher of a multi-stage tactical air war game will be reviewed in Chapter VI.

The Lulejian Model

The Lulejian model was developed for the Weapon Systems Evaluation Group (WSEG) by Lulejian and Associates, Inc. in 1973 (Ref 10).

The model employs an algorithm that finds the appropriate game value and approximate optimal strategies for both players for each stage of the game (i.e., campaign). The basic approach is to decompose the problem into the solution of many one-move games. The discussion of the algorithm appearing below was obtained from Reference 10.

> The vector $x = (x_1, \ldots, x_m)$ will denote the state of the game where m is the number of state variables. . . . Let A and B be the sets of strategies available to players 1 and 2, respectively, on each move. . . . The incremental payoff function, the terminal payoff function, and the transition function will be called g, h, and t, respectively. Thus, if the current state is x, and the strategies chosen by the players are a and b, then the incremental payoff for the move is $g(x,a,b)$ and the state x' at the beginning of the next move will be $x' = t(x,a,b)$. Finally, let $V_N(x)$ denote the game value of an N-move game which is started at the initial state x and played with optimal strategies by both players.
>
> An analytic solution of an N-move game is a function, V_N, giving the game value as a function of the initial state. Such solutions are rare because of formidable mathematical difficulties. For games which possess an analytic solution the procedure is as follows:
>
> First, the one-move game is solved:
>
> $$V_1(x) = \min_{b \varepsilon B} \max_{a \varepsilon A} [g(x,a,b)]$$
>
> Then using the one-move solution, the two-move game is solved:
>
> $$V_2(x) = \min_{b \varepsilon B} \max_{a \varepsilon A} [g(x,a,b) + V_1(t(x,a,b))]$$
>
> This recursive procedure is continued, one move at a time, until finally V_N is reached
>
> $$V_N(x) = \min_{b \varepsilon B} \max_{a \varepsilon A} [g(x,a,b) + V_{N-1}(t(x,a,b))]$$

. . . the . . . games typically do not possess saddle points. Thus, pure strategy solutions to the above minimax problem do not exist. Rather than going to mixed strategy solutions, the Lulejian model circumvents this difficulty by computing the game value for one player or the other announcing his strategy choice first. These values provide upper and lower bounds for the simultaneous move (mixed strategy) game value.

Let $V_k^{(1)}(x)$ be the game value for player 1 (the maximizer) announcing his strategy first. Then

$$V_k^{(1)}(x) = \max_{a\epsilon A} \left[\min_{b\epsilon B} [P_k(x,a,b)]\right] ,$$

where

$$P_k(x,a,b) = g(x,a,b) + V_{k-1}(t(x,a,b)) .$$

That is, player 1 will announce that strategy which will maximize the minimum payoff he will receive regardless of player 2's subsequent strategy choice. Similarly, let $V_k^{(2)}(x)$ be the game value for player 2 (the minimizer) announcing his strategy first. Then

$$V_k^{(2)}(x) = \min_{b\epsilon B} \left[\max_{a\epsilon A} [P_k(x,a,b)]\right]$$

It can be shown that for any x

$$V_k^{(1)}(x) \leq V_k^{(2)}(x)$$

Furthermore, if $V_k(x)$ is the simultaneous move game value, then

$$V_k^{(1)}(x) \leq V_k(x) \leq V_k^{(2)}(x)$$

Suppose player 2 announces his strategy first. Then, the algorithm employed in the prototype model proceeds as follows. In the expression

$$V_k^{(2)}(x) = \min_{b\epsilon B} \left[\max_{a\epsilon A} [g(x,a,b) + V_{k-1}^{(2)}(t(x,a,b))]\right]$$

the function $V_{k-1}^{(2)}$, which represents the value of the remaining moves in the game, is approximated by playing the game through to the end using strategies which have been previously optimized. The procedure is as follows:

1. Estimate good strategies for both players for the game. Let $(a_N,b_N),\dots,(a_1,b_1)$ be the estimate.

2. Find the state path $x^{(N)},\dots,x^{(1)}$ determined by these strategies and the initial state $x^{(N)}$. [The initial state is given; $x^{(1)}$ represents the state of the beginning of the stage.]

3. Find the total payoff P_N.

4. Optimize the strategies (a_1,b_1) on the last move, i.e., calculate $V_1^{(2)}(x^{(1)})$. Let the new strategies be (a_1',b_1').

5. Optimize the next to last move strategies (a_2,b_2) by assuming the value of the last move is approximately equal to the incremental payoff when the optimized strategies (a_1',b_1') are used. (The assumption is that the relative value of various next to the last move strategies can be seen by playing the last move with an optimized but fixed last move strategy.) Let the new strategy be (a_2',a_2').

6. Having found the optimized strategies $(a_{k-1}',b_{k-1}'),\dots,(a_1',b_1')$, optimize (a_k,b_k) by approximating $V_{k-1}^{(2)}$ by the payoff obtained by playing the remaining k-1 moves using the previously optimized strategies. Repeat until k=N.

7. Compare the new strategies $((a_N',b_N'),(a_{N-1}',b_{N-1}'),\dots(a_1',b_1'))$ with the old strategies. If they are identical, then the algorithm has converged. The game value is approximately P_N' and the strategies found in (6) are approximately optimal strategies for the two players. If the strategies do not agree, go to step (2) and repeat the process until convergence is obtained.

The sequence in which the strategies are optimized from the last move (k=1) to the first move (k=N) is actually quite arbitrary. The reverse sequence may just as well be used or, better, the two sequences in alternation.

OPTSA

The OPTimal Sortie Allocation model, OPTSA II, was developed by IDA for computing the percentage of assignments of general-purpose aircraft to missions by period, for up to three periods. It is the only technique that provides an exact solution but it is very time-consuming (Ref 2). The description below for the game solution procedure was obtained from Reference 2.

Let $u(i_B, i_R, j_B, j_R, k_B, k_R)$ denote the outcome of the game in the selected measure of effectiveness if Blue chooses action i_B, j_B, k_B on moves 1,2,3, and Red chooses action i_R, j_R, k_R on moves 1,2,3 where $i_B = 1,\ldots,I_B$; $j_B = 1,\ldots,J_B$; $k_B = 1,\ldots,K_B$ and $i_R = 1,\ldots I_R$; $j_R = 1,\ldots,J_R$ and $k_R = 1,\ldots,K_R$.

Let $v(i_B, i_R, j_B, j_R)$ denote the expected value of the selected measure of effectiveness if Blue chooses action i_B, j_B on moves 1,2, and Red chooses action i_R, j_R on moves 1,2 and then both play optimally on move 3.

Let $w(i_B, i_R)$ denote the expected value of the selected measure of effectiveness if Blue chooses action i_B on move 1 and Red chooses action i_R on move 1 and then both play optimally on moves 2 and 3.

Let V denote the expected value of the game with optimal plays by both Red and Blue on all three moves.

The value $v(i_B, i_R, j_B, j_R)$ is the expected value of a mixed strategy on the third move. The probabilities can be denoted by

$$p^3(i_B, i_R, j_B, j_R, k_B), \quad \text{all } i_B, i_R, j_B, j_R, k_B$$

$$q^3(i_B, i_R, j_B, j_R, k_R), \quad \text{all } i_B, i_R, j_B, j_R, k_R.$$

The value $w(i_B, i_R)$ is the expected value of a mixed strategy on the second move. The probabilities can be denoted by

$$p^2(i_B, i_R, j_B), \quad \text{all } i_B, i_R, j_B$$

$$q^2(i_B, i_R, j_R), \quad \text{all } i_B, i_R, j_R.$$

The value V is the expected value of a mixed strategy on the first move. The probabilities can be denoted by

$$p^1(i_B), \quad \text{all } i_B$$

$$q^1(i_R), \quad \text{all } i_R.$$

For each combination of values i_B, i_R, j_B, j_R, the value of the outcome $u(i_B, i_R, j_B, j_R, k_B, k_R)$ is computed for $k_B = 1,\ldots,K_B$ and $k_R = 1,\ldots,K_R$. Then the game consisting of these values is solved to obtain a value $v(i_B, i_R, j_B, j_R)$ and associated optimal third move mixed strategies $p^3(i_B, i_R, j_B, j_R, k_B)$, $k_B = 1,\ldots,K_B$ and $q^3(i_B, i_R, j_B, j_R, k_R)$, $k_R = 1,\ldots,K_R$.

For each i_B, i_R the game consisting of values $v(i_B, i_R, j_B, j_R)$ is solved to obtain a value $w(i_B, i_R)$ and an associated optimal second-move mixed strategies $p^2(i_B, i_R, j_B)$, $j_B = 1,\ldots,J_B$ and $p^2(i_B, i_R, j_R)$, $j_R = 1,\ldots,J_R$.

Finally the game consisting of values $w(i_B, i_R)$ is solved to obtain V and associated optimal first-move mixed strategy $p^1(i_B)$, $i_B = 1,\ldots,I_B$ and $p^1(i_R)$, $i_R = 1,\ldots,I_R$.

If the game is three-move, six-action there are at the first computation stage above 1296 6 x 6 games with resulting values $v(i_B, i_R, j_B, j_R)$ $i_B = 1,\ldots,6$, $i_R = 1,\ldots,6$, $j_B = 1,\ldots,6$, $j_R = 1,\ldots,6$. At the second computation stage there are 36 6 x 6 games with resulting values $w(i_B, i_R) i_B = 1,\ldots,6$, $i_R = 1,\ldots,6$. At the third computation stage there is one 6x6 game with resulting value V.

For each of the 1296 games there is a mixed strategy for Blue and Red, consisting of six probabilities. For each of the 36 games there is a mixed strategy. For the one game there is a mixed strategy. Thus there are $1296 + 36 + 1 = 1333$ games, each with a value and a mixed strategy for Blue and Red.

For input to each of the 1296 games there must be computed 36 evaluations of . . . [outcomes]. Therefore 46,656 evaluations . . . are necessary.

If there were 10 actions at each move, there would be 100 states after move 1, $100 \times 100 = 10{,}000$ states after move 2, and $10{,}000 \times 100 = 1{,}000{,}000$ states after move 3. Thus roughly 22 times as much computation is required to obtain the outcomes. The number of games is $10{,}000 + 100 + 1 = 10{,}101$, or roughly 8 times as many games to solve (10 x 10 games rather than 6 x 6 games).

As an example, consider a 2 x 2 game with the payoff matrix for the Blue force shown in Table IV-1. b represents the amount of a resource available to blue at the start of the stage. The state transition function describing how the strategies affect the amount of b available at the next stage is contained in Table IV-2.

TABLE IV-1

PAYOFF MATRIX

		RED	
		1	2
BLUE	1	5(b)	3(b)
	2	2(b)	4(b)

TABLE IV-2

2x2 STATE TRANSITION MATRIX

		RED	
		1	2
BLUE	1	.4(b)	.2(b)
	2	.5(b)	.3(b)

Suppose the game begins with b=1. Thus, if both sides play strategy 1 on their first play, then blue receives a payoff of 5 and the value of b entering the next stage is .4. If again both sides select strategy 1, then blue receives an additional payoff 5(.4) or 2 for a total payoff to date of 5+2 or 7. The value of b entering the last (third) stage is (.4)(.4) or .16. If both sides again select strategy 1, then blue receives an additional payoff of .16(5) or .8. The game is over and the total payoff achieved by blue is 7.8.

<u>Step 1</u>. To solve this game using OPTSA we first create a table containing the total payoff values for all 64 different strategy options. A partial listing of this Table is given as Table IV-3.

TABLE IV-3

TOTAL GAME PAYOFF VALUES FOR DIFFERENT STRATEGY SELECTIONS IN 2x2 GAME

First Stage		Second Stage		Third Stage		Total Payoff
i_B	i_R	j_B	j_R	k_B	k_R	
1	1	1	1	1	1	7.8
1	1	1	1	1	2	7.48
1	1	1	1	2	1	7.32
1	1	1	1	2	2	7.64
1	1	1	2	1	1	6.6
1	1	1	2	1	2	6.44
1	1	1	2	2	1	6.36
1	1	1	2	2	2	6.52
⋮		⋮		⋮		⋮
2	2	2	2	2	2	5.56

<u>Step 2</u>. Second, a table containing the values of $V(i_B, i_R, j_B, j_R)$ is created. Recall that $V(i_B, i_R, j_B, j_R)$ is the expected payoff if Blue chooses i_B, j_B on moves 1,2 and Red chooses i_R, j_R on moves 1,2 and then both play optimally on move 3. There will be 16 entries in this table. Each one is found by solving a 2x2 game. For example, to obtain V(1,1,1,1) we solve the 2x2 game (shown

in Table IV-4) where the individual payoffs are obtained from Table IV-3.

TABLE IV-4

2x2 GAME FOR V(1,1,1,1)

K_B \ K_R	1	2
1	7.8	7.48
2	7.32	7.64

The total payoff for this game is 7.56 (solution technique discussed in Chapter II). A partial listing of the table for $V(i_B, i_R, j_B, j_R)$ appears in Table IV-5.

TABLE IV-5

VALUES OF $V(i_B, i_R, j_B, j_R)$ FOR 2x2 GAME

First Stage		Second Stage		
i_B	i_R	j_B	j_R	$V(i_B,i_R,j_B,j_R)$
1	1	1	1	7.56
1	1	1	2	6.48
1	1	2	1	6.50
1	1	2	2	7.02
1	2	1	1	5.28
⋮		⋮		⋮
2	2	2	2	5.51

<u>Step 3</u>. Third, a table is constructed containing the values of $W(i_B, i_R)$, the expected total payoff when Blue chooses i_B and Red chooses i_R on move 1 and then both play optimally on moves 2 and 3. To find W(1,1) we must solve the following game (Table IV-6), with individual payoffs obtained from Table IV-5.

TABLE IV-6

2x2 GAME FOR W(1,1)

J_B \ J_R	1	2
1	7.56	6.48
2	6.50	7.02

The game value and hence W(1,1) is 6.845. Table IV-7 summarizes the results.

TABLE IV-7

$W(i_B, i_R)$ FOR 2x2 GAME

First Stage		
i_B	i_R	$W(i_B, i_R)$
1	1	6.845
1	2	3.922
2	1	4.306
2	2	5.383

Step 4. Finally we obtain the optimal expected payoff for the game by solving the following game (Table IV-8) with payoff entries obtained from Table IV-7.

TABLE IV-8

2x2 GAME FOR OPTIMAL SOLUTION

i_R \ i_R	1	2
1	6.845	3.922
2	4.306	5.383

The optimal game value is 4.990. The corresponding (perhaps mixed) strategies can be found by considering the individual game solutions that contributed to the final payoff. Note that 16+4+1 or 21 (2x2) games were solved in the OPSTA solution procedure.

DYGAM

DYGAM (DYnamic GAMes solver) was developed at Control Analysis Corporation to solve general multi-stage games. The game solution procedure is as follows (Ref 6).

Let $W=(W_i)$ represent a N-dimensional vector of state variables which completely characterizes the status of the game at the beginning of any step. Assume that there exists a known payoff function $p^{(T+1)}(W)$ which gives the reward to the Blue side if the game ends (at the end of the T^{th} stage) in state W. Also assume that the payoff to the Red side is $-p^{(T+1)}(W)$, so that the game is zero-sum. Define $p^{(1)}(W)$ to be the optimal payoff (if both sides make their best moves) to Blue if the initial state is equal to W at the end of the game. Thus, $-p^{(1)}(W)$ will be the optimal payoff to Red. The problem is to compute $p^{(1)}(W)$ for any initial deployment, W, and in this way evaluate alternative initial deployments.

Rather than directly computing the optimal payoff $p^{(1)}(W)$, the dynamic programming approach first computes the intermediate payoffs $p^{(t)}(W)$, for t=T, T-1, T-2, etc. Here, $p^{(t)}(W)$ has the interpretation of being the optimal payoff to Blue when starting from state W at the beginning of the t^{th} period. The method begins by determining the best strategies during the last or T^{th} stage. For example, suppose that the state of the game was W at the beginning of the T^{th} stage. Then only a one-stage problem need be solved to determine the value $p^{(T)}(W)$ of the game. If a simultaneous game is being played (both sides redeploy during each stage), then a single stage game must be solved, perhaps by linear programming. . . . Unfortunately, it will be necessary in general to determine or estimate $p^{(T)}(W)$ for all possible states W. The next step is to determine the strategies during the second to last stage (stage T-1). Again, only a one-stage problem need be solved to determine the value $p^{(T-1)}(W)$; here, $p^{(T)}(\cdot)$ is used to compute the relative value of states at the beginning of the T^{th} period. This process is repeated until the first stage is reached, where the function $p^{(t)}(\cdot)$ is computed from the function $p^{(t+1)}(\cdot)$ for t=1,...,T by solving a single stage optimization problem (a single stage game for simultaneous games.

. . . Note that the optimization proceeds backwards in time, starting from $p^{(T+1)}(\cdot)$ and ending with $p^{(1)}(\cdot)$.

In dynamic programming it is generally necessary to determine (or estimate) $p^{(t)}(W)$, $t=1,\ldots,T$, for all possible states W. Rather than determining these payoff functions exactly, the approach taken is to use a polynomial with sufficient degree to represent this function. According to the Polynomial Approximation Theorem, there exists a polynomial function which can approximate any continuous function over a compact set within any specified accuracy. Basically, there are two approaches for fitting a polynomial to a function. One is to use a single polynomial with high degree to approximate the payoff function over the entire state space. . . . The second method is to subdivide the original space into smaller regions and then use a separate polynomial of lower order to approximate the payoff function over each region; this latter approach is called subregional approximation and is used by DYGAM.

V. The Algorithm

Introduction

The purpose of this chapter is to carefully develop a general algorithm which can be used to solve multi-stage two-person zero-sum sumultaneous games. The basic approach to solve the game is to decompose the multi-stage game into the solution of many one-stage games. The main idea is to build at each stage a matrix whose values are the payoffs obtained by playing each of the given strategies at this stage and the optimal strategies for the remaining stages. A dynamic programming (backward approach) is used to determine the payoff obtained by using the optimal strategies for the remaining stages.

The Algorithm

In order to facilitate the discussion of the solution technique, some notation will be necessary.

Let B and R be the number of strategies available to the Blue and Red players, respectively, at each stage. It is convenient to count stages from the end of the campaign to the beginning. Thus ISTAG = 1 will represent the last stage, the final battle and ISTAG = MSTAGE will represent the beginning of the battle, where MSTAGE is the total number of stages.

The vector $I^{(ISTAG)} = (x_1, x_2, \ldots, x_N)$ will denote one possible state of the game at stage ISTAG where N is the number of state variables. For example, if the state of the game at a stage is characterized by the ratio of forces (x_1), then N=1 and $I^{(ISTAG)} = (x_1)$.

Assume that there exist known payoff matrices with elements $TT(I^{(ISTAG)},J,K)$ and $-TT(I^{(ISTAG)},J,K)$ which gives the reward (payoff) to the Blue side and the Red side, respectively, if Blue played his Jth strategy and Red played his Kth strategy at the stage ISTAG with entering state vector I^{ISTAG}. Assume too that there exist a known state transition function

$$I^{(ISTAG-1)} = F(I^{(ISTAG)},J,K)$$

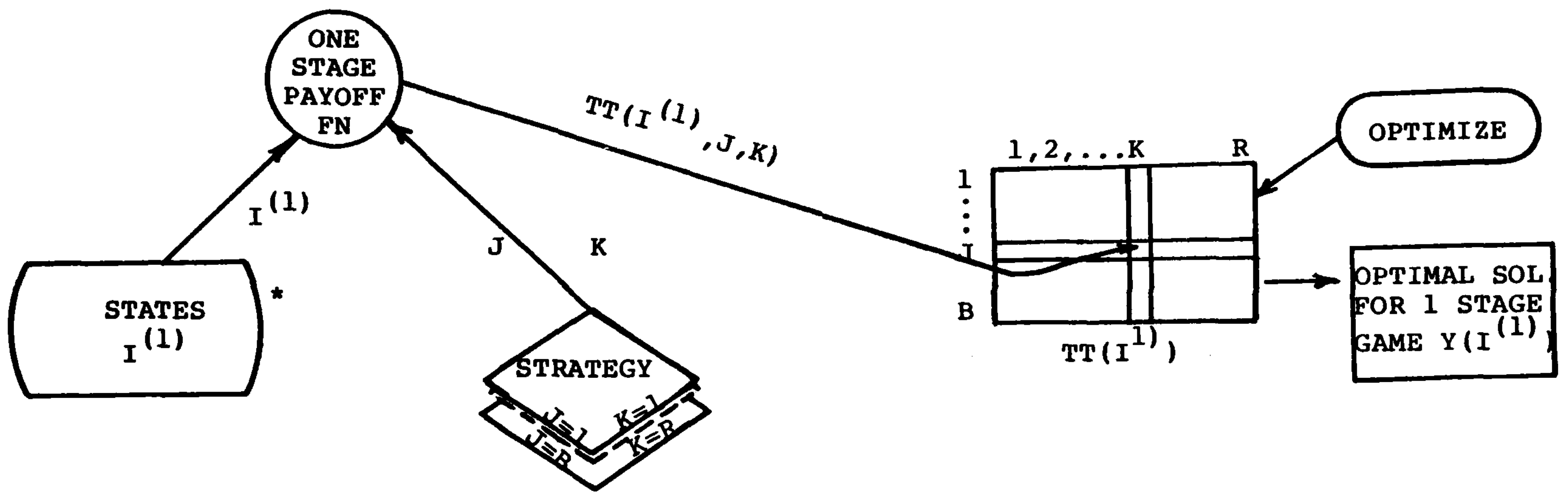
which gives the value of the state vector $I^{(ISTAG-1)}$ at the beginning of the previous stage from the state vector for this stage and the players' strategies at this stage.

Define $Y(I^{(ISTAG)})$ to be the optimal payoff to Blue if both sides make their best moves from ISTAG until the final battle when the state vector equals I^{ISTAG} at ISTAG. The problem is to compute $Y(I^{(MSTAGE)})$ for any initial state, $I^{(MSTAGE)}$, and total number of stages MSTAGE. For example, consider a five-stage game with an initial Blue force of BLUF = 150 and Red force of REDF = 100. Again assume a state vector consisting of a single variable, namely the force ratio. Then it is required to compute $Y(I^{(5)}$, where $I^{(5)} = (x_1) = 150/100 = 1.5$.

Rather than directly computing the optimal payoff $Y(I^{(MSTAGE)})$, the dynamic programming approach first computes the intermediate payoffs $Y(I^{(ISTAG)})$ for ISTAG = 1, 2, ... MSTAGE - 1 and for each $I^{(ISTAG)}$. The method begins by determining the optimal strategies during stage 1 for all possible values of the state variable $I^{(ISTAG)}$ (Figure V-1). For example, suppose that the game was at state $I^{(1)}$ at the beginning of stage 1. Then only the one-stage problem with payoff matrix $TT(I^{(1)})$ containing elements $TT(I^{(1)}, J, K)$, $J = 1,...B, K=1,...R$, need be solved to determine the value $Y(I^{(1)})$. For the simultaneous game (both sides redeploy during each stage) a single-stage problem can be solved by linear programming (see Chapter II). The game value of this matrix will be $Y(I^{(1)})$. As mentioned earlier, in theory it will be necessary to determine $Y(I^{(1)})$ for all possible states $I^{(1)}$ (e.g., for all possible force ratios that could enter the final state). If $I^{(1)}$ is very large, then this can be a very time-consuming task. Later in this chapter an approximation technique is discussed which lessens this computational burden.

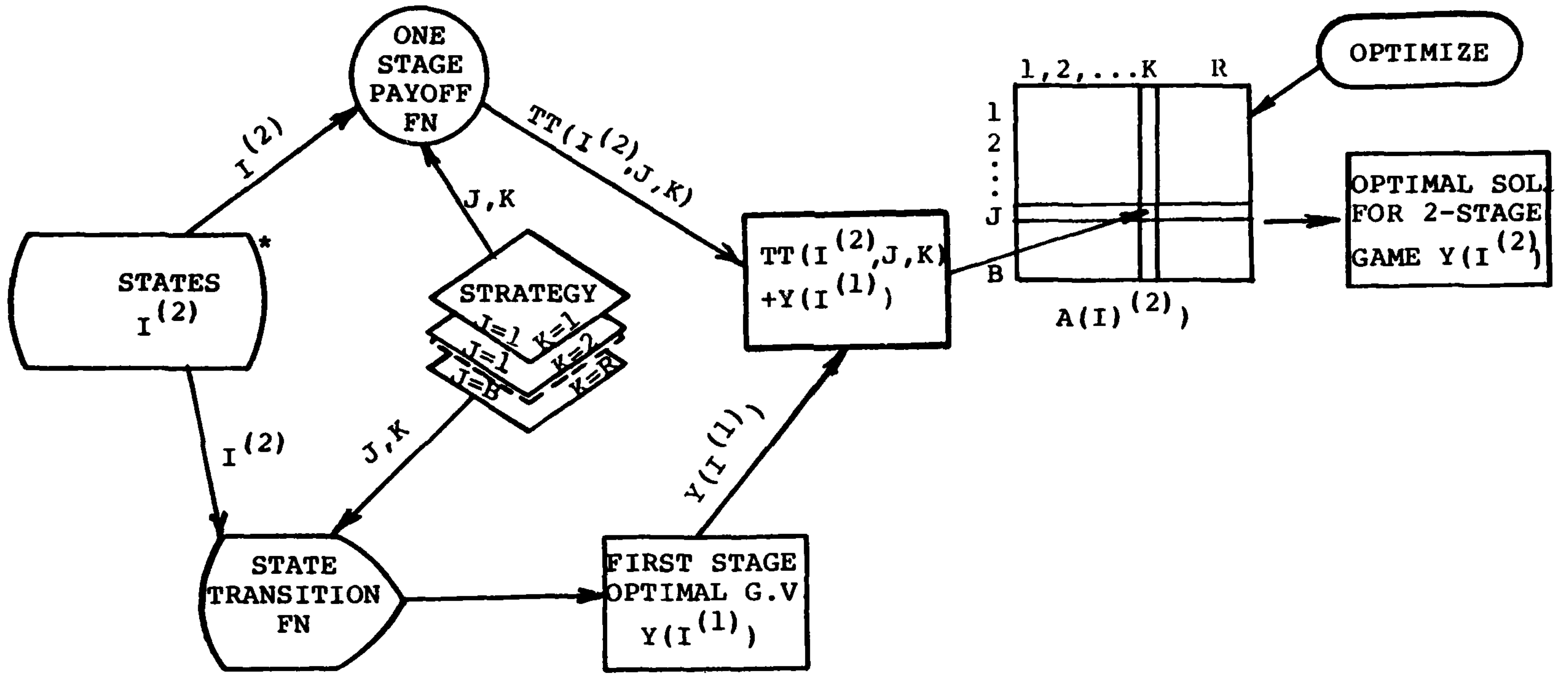
The algorithm progresses through the following steps (Figure V-2):

1. For each possible state vector $I^{(2)}$ at the beginning of the second stage and strategy pairs J and K



* Process repeated for all possible values of the state vector $I^{(1)}$.

Fig. V-1. Generating the Optimal Solution for a One-Stage Game



*Repeated for all possible values of the state vector $I^{(2)}$

Fig. V-2. Generating the Optimal Solution for a Two-Stage Game

for Blue and Red, respectively, use the state transition function to calculate the values in the entering state vector $I^{(1)}$ for the first stage.

2. The optimal game value $Y(I^{(1)})$ for the state vector entering the first stage has already been computed. This value is added to the payoff value, $TT(I^{(2)},J,K)$ for the second stage. This sum $(A(I^{(2)},J,K))$ is the payoff to Blue if Blue uses strategy J, Red uses strategy K at the next to last stage (ISTAG = 2) and then both use their otpimal strategies at the last stage (ISTAG = 1).

3. Let $A(I^{(2)})$ be the payoff matrix obtained when the state vector is $I^{(2)}$ at ISTAG = 2, then this one-stage game can be solved to provide the optimal solution for the two-stage game. The solution to the two-stage game is denoted by $Y(I^{(2)})$.

4. Repeat until ISTAG = MSTAG.

In dynamic programming, it is generally necessary to determine $Y(I^{(ISTAG)})$, ISTAG = 1, 2, ..., MSTAGE, for all possible states I^{ISTAG}. Rather than determining the game values for all possible states, the approach taken is to use a suitable number of grid points selected at a suitable distance over the output state space of any stage. For example, if the possible force ratios are between .1 and 10, the grid points can be selected so the values of the state vector $I^{(ISTAG)}$ considered are {.1, .2, .4, .7, 1, 1.5, 2, 4, 6, 8, 10}. A cubic spline interpolation

(Appendix A) is then used to estimate the payoffs for force ratios between the selected grid points.

VI. Tactical Air Application

Introduction

The remainder of this thesis will be concerned with the application of the methodology developed in Chapter V to tactical air warfare problems. This chapter contains a discussion of the missions in tactical air warfare and indicates the major assumptions necessary to formulate the problem as a multi-stage two-person zero-sum game.

The Problem

The problem of optimal employment of a tactical air force in various theater air missions can be analyzed as a multi-stage game between two sides. Tactical air forces may be used on many missions, such as the following:

Air Base Attack. These operations are against the enemy's theater air base complex and organization with the purpose of destroying his aircraft, personnel, facilities, and so on.

Air Defense. These represent air-defense operations against the enemy's air base attack operations.

Close Air Support. The targets for close air support operations are concentrations of enemy troops or fortified positions close to the FEBA. They are attacked in order to help the ground forces in the battle area.

Interdiction. These operations reduce the enemy's military potential by attacking his transportation facilities.

Reconnaissance. The most important function of these operations is to obtain information about enemy targets.

By setting three categories--attack, defense, and support--it is apparent that each of the five tasks just mentioned can be placed into one or more of these three categories (Ref 11:233). For example, counter air would go into the attack category. Air defense would be placed under defense, and close air support under the support category. Both reconnaissance and interdiction could go into the categories of attack and support. Thus the problem of tactical air war becomes the problem of employing the tactical air force in the three missions of attack, defense, and support for each stage of the war.

Assumptions

Since the ultimate objective is to win the ground battle, we will assume each player's objective for the game is to maximize the difference between his close air support and his opponent's close air support for the entire campaign. The players make allocation decisions between airborne attack, air defense and close air support each day; these determine the outcome of the day's encounter. The decision is not easy. Large allocations to air base attack early in the campaign may result in substantially degrading the enemy's ability to conduct close air support later in the campaign. The next day a new game is played. The sequential total game for the entire campaign is referred to as N-stage game, where N is the number of time periods (days) represented in the campaign.

Additional assumptions of the model include:

1. Initial friendly and enemy force size and characteristics are fixed and the campaign progresses to completion without force replenishment by either side.

2. The campaign consists of "simultaneous" stages. Redeployment of aircraft is made by both sides at each stage. Each side knows both his strength and the strength of the opposing force. However, neither side knows the other's deployment.

3. Aircraft are assumed to be homogeneous and of multiple capability. Thus any available aircraft may be assigned to a counter air mission, air defense mission or close air support mission.

4. Attacks on air bases are limited to destruction of aircraft caught on the ground. No benefit is assumed to occur from these attacks on the installation, POL, personnel, etc. Initially it is assumed all of the

opposing player's aircraft are on the ground at the time of attack.

5. Aircraft engaging in close air support cannot be destroyed when accomplishing their missions. Thus there is no attrition due to ground forces (AAA, SAMS) or air defenses at the battle field.

6. Air defense aircraft are assumed to cause a portion of the attacking force to jettison ordnance and return to base. Initially it is assumed that the air defense aircraft do not destroy the enemy attackers. Later, in the binomial model, this assumption is relaxed.

Summary

In this chapter the tactical air war tasks were compressed into three missions--attack, defense and close air support. In addition, it was shown how the tactical air war game could be formulated as a multi-stage, two-person zero-sum simultaneous game. In the next chapter a linear model is developed to solve the game. For the linear model, all aircraft attrition is the result of an air base attack on the other side. In Chapter IX a binomial model is developed which allows for both air-to-air attrition and ground attrition.

VII. Formulation and Solution of the Linear Model

Introduction

In this chapter we assume that aircraft attrition can only occur due to aircraft assigned to the air base attack mission destroying aircraft at the air base. Each aircraft attacking the base is assumed to destroy a fixed number of aircraft on the ground. Each aircraft assigned to air defense causes a fixed number of air base attackers to abort their mission and return to their home base. Thus the number of air base attack aircraft that abort their mission is a linear function of the number of air defense aircraft. Hence the name linear model.

The first section of this chapter provides a detailed formulation of the linear model. The next section provides the linear model solution. This section is divided into subsections dealing with the problem of an infinite number of stages, then one stage equations for the linear model, and the dynamic programming approach used to solve the problem. Finally, a technique that allows the algorithm to obtain an approximate solution when the state variable is continuous is discussed.

Formulation of the Linear Model

The linear model requires each player to allocate all available aircraft to each of the three missions--air base attack, air defense, and close air support--for each stage of the game. Let $\mathrm{BLUF}^{(\mathrm{ISTAG})}$ be the number of Blue aircraft at the beginning of stage ISTAG, and $\mathrm{REDF}^{(\mathrm{ISTAG})}$ be the number of Red aircraft at the beginning of ISTAG stage. These are the state variables in the model. The decision variables are: $\mathrm{ABAB}^{(\mathrm{ISTAG})}$ is the number of Blue aircraft sent on air base attack during stage ISTAG; $\mathrm{ADB}^{(\mathrm{ISTAG})}$ is the number of Blue aircraft sent on air defense operations; $\mathrm{ABAR}^{(\mathrm{ISTAG})}$ is the number of Red aircraft sent on air base attacks; and $\mathrm{ADR}^{(\mathrm{ISTAG})}$ is the number of Red aircraft sent on air defense operations. The remaining number of Blue aircraft,

$$\mathrm{CASB}^{(\mathrm{ISTAG})} = \mathrm{BLUF}^{(\mathrm{ISTAG})} - \mathrm{ABAB}^{(\mathrm{ISTAG})} - \mathrm{ADB}^{(\mathrm{ISTAG})} \tag{7-1}$$

is the number sent on close air support operations; and the remaining number of Red planes,

$$\mathrm{CASR}^{(\mathrm{ISTAG})} = \mathrm{REDF}^{(\mathrm{ISTAG})} - \mathrm{ABAR}^{(\mathrm{ISTAG})} - \mathrm{ADR}^{(\mathrm{ISTAG})} \tag{7-2}$$

is the number sent in close air support operations by Red. This is a simultaneous game, as both Red and Blue allocate their available aircraft to the various missions during every stage of the campaign.

Assume that each Red defender is able to engage on the average with RAD Blue attackers and force them to jettison their load and abort the mission. So the number of Blue aircraft which can complete the air base attack mission will be the

$$\max(0, \mathrm{ABAB}^{(\mathrm{ISTAG})} - \mathrm{RAD} * \mathrm{ADR}^{(\mathrm{ISTAG})})$$

Assume that the fraction of Red aircraft not caught on the ground is FRNC, and that each Blue attacker can destroy on the average BABA Red aircraft caught on the ground. Then the inventory of Red aircraft at the beginning of the next stage is[1]

$$\mathrm{REDF}^{(\mathrm{ISTAG-1})} = \max[\mathrm{REDF}^{(\mathrm{ISTAG})} \times \mathrm{FRNC},\ \mathrm{REDF}^{(\mathrm{ISTAG})} - \mathrm{BABA} \times \max(0, \mathrm{ABAB}^{(\mathrm{ISTAG})} - \mathrm{RAD} \times \mathrm{ADR}^{(\mathrm{ISTAG})}]; \tag{7-3}$$

and the inventory of Blue aircraft at the beginning of the stage ISTAG-1 is

$$\mathrm{BLUF}^{(\mathrm{ISTAG-1})} = \max[\mathrm{BLUF}^{(\mathrm{ISTAG})} \times \mathrm{FBNC},\ \mathrm{BLUF}^{(\mathrm{ISTAG})} - \mathrm{RABA} \times \max(0, \mathrm{ABAR}^{(\mathrm{ISTAG})} - \mathrm{BAD} \times \mathrm{ADB}^{(\mathrm{ISTAG})} \tag{7-4}$$

where FBNC is the fraction of Blue aircraft not caught on the ground, RABA is the expected number of Blue aircraft

[1]Recall that stages are counted from the final battle to the first battle. Hence ISTAG-1 is later in time than ISTAG.

destroyed on the ground by a Red attacker that successfully attacks the base, and RAD is the expected number of Red attacker forced by one Blue defender to abort their mission.

The payoff to Blue for an entire campaign of MSTAGE stages is given by

$$M = \sum_{ISTAG=1}^{MSTAGE} [BCAS \times CASB^{(ISTAG)} - RCAS \times CASR^{(ISTAG)}] \quad (7\text{-}5)$$

where BCAS, RCAS are the Blue and Red aircraft close air support capabilities, respectively. Thus, the problem that the Blue decision maker (Red decision maker) faces is to determine the aircraft allocation for each stage so as to maximize (minimize) Equation (7-5) subject to Equations (7-1), (7-2), (7-3), and (7-4).

Linear Model Solution

The Problem of an Infinite Number of Stages.

The game formulated in the preceding section is a continuous game. That is, the force allocation ratios, $ABAB^{(ISTAG)}/BLUF^{(ISTAG)}$, $ADB^{(ISTAG)}/BLUF^{(ISTAG)}$, $ABAR^{(ISTAG)}/REDF^{(ISTAG)}$, and $ADR^{(ISTAG)}/REDF^{(ISTAG)}$ may be chosen to be any real number within the interval from zero to one. Thus there are an infinite number of possible allocations for each stage. In the algorithm developed in Chapter V, on the other hand, it is assumed that there are only a finite number of possible strategies at each stage. Thus it was necessary to limit the strategy options in the following way: the ratios (fractions) $ABAB^{(ISTAG)}/BLUF^{(ISTAG)}$, $ADB^{(ISTAG)}/BLUF^{(ISTAG)}$, $ABAR^{(ISTAG)}/REDF^{(ISTAG)}$, and $ADR^{(ISTAG)}/REDF^{(ISTAG)}$ must be non-negative integer multiples of 1/M where M is a positive integer greater than one. For example, if M=3, then $ABAB^{(ISTAG)}$ would be either 0, $BLUF^{(ISTAG)}/3$, $2\,BLUF^{(ISTAG)}/3$, or $BLUF^{(ISTAG)}$. With this notation, the total number of strategies that each side has during each stage is given by the possible number of ways M balls can be allocated to three boxes. This is $\binom{M+3-1}{3-1} = \frac{(M+2)(M+1)}{2}$. In the test runs, M=3 and M=5 were used.

Let s(J,L) represent the fraction of the Blue force allocated in CAS (L=1), ABA (L=2), and AD (L=3) for all available blue strategies J, $(J=1, \frac{(M+2)(M+1)}{2})$. Let s(K,L), L=1,2,3, represent the ratio of the Red force allocated in CAS, ABA, AD respectively for all available Red strategies K, $(K=1,\frac{(M+2)(M-1)}{2})$. For example, if M=3 both the matrices s(J,L), s(K,L) will be as shown in Table VII-1.

<u>One Stage Equations of the Linear Model</u>. Given for any engagement between Blue and Red with force sizes $BLUF^{(ISTAG)}$, $REDF^{(ISTAG)}$, where the Blue has a fraction s(J,1) of his force assigned to the close air support,

TABLE VII-1

STRATEGIES AVAILABLE FORCE ALLOCATION AT M=3

STRATEGY	1	2	3	4	5	6	7	8	9	10
CAS	3/3	2/3	2/3	1/3	1/3	1/3	0	0	0	0
ABA	0	1/3	0	2/3	1/3	0	3/3	2/3	1/3	0
AD	0	0	1/3	0	1/3	2/3	0	1/3	2/3	3/3

the fraction s(J,2) assigned to air base attack and the fraction s(J,3) assigned to air defense where

$$\sum_{L=1}^{3} s(J,L) = 1, \qquad s(J,L) \geq 0, \ L = 1,2,3.$$

The Red has a fraction s(K,1), s(K,2) and s(K,3) assigned to close air support, air base attack and air defense respectively. The payoff for a single stage of the game with Blue playing his Jth strategy and Red playing his Kth strategy will be

$$\begin{aligned} \text{PAYOFF}^{(\text{ISTAG})} &= \text{BCAS} \times \text{BLUF}^{(\text{ISTAG})} \times s(J,1) \\ &\quad - \text{RCAS} \times \text{REDF}^{(\text{ISTAG})} \times s(K,1 \\ &= \text{REDF}^{(\text{ISTAG})} \ [\text{BCAS} \times \frac{\text{BLUF}^{(\text{ISTAG})}}{\text{REDF}^{(\text{ISTAG})}} \times s(J,1) \\ &\quad - \text{RCAS} \times s(K,1)] \qquad (7\text{-}6) \end{aligned}$$

Equations (7-3) and (7-4) for the force size of Blue and Red at the beginning of stage ISTAG-1 can be re-expressed as

$$\begin{aligned} BLUF^{(ISTAG-1)} = \max(&BLUF^{(ISTAG)} \times FBNC,\ BLUF^{(ISTAG)} \\ &- RABA \times \max[0, REDF^{(ISTAG)} \times s(K,2) \\ &- BAD \times BLUF^{(ISTAG)} \times s(J,3)]) \end{aligned} \tag{7-7}$$

$$\begin{aligned} REDF^{(ISTAG-1)} = \max(&REDF^{(ISTAG)} \times FRNC,\ REDF^{(ISTAG)} \\ &- BABA \times \max[0, BLUF^{(ISTAG)} \times s(J,2) \\ &- RAD \times REDF^{(ISTAG)} \times s(K,3)]) \end{aligned} \tag{7-8}$$

We can use $BLUF^{(ISTAG)}$ and $REDF^{(ISTAG)}$ as state variables and find all possible values of these variables at each stage by using the payoff function given in Equation (7-6) and the transition functions given in Equations (7-7) and (7-8).

But this procedure can be simplified. First let $BI^{(ISTAG)}$ be the ratio of the Blue and Red forces at stage ISTAG. Then

$$BI^{(ISTAG)} = \frac{BLUF^{(ISTAG)}}{REDF^{(ISTAG)}}$$

Hence the state variables $REDF^{(ISTAG)}$ and $BI^{(ISTAG)}$ contain the same information as the original state variables $REDF^{(ISTAG)}$ and $BLUF^{(ISTAG)}$.

Second, define $RR^{(ISTAG)}$ as the ratio of successive Red force sizes.

$$RR^{(ISTAG)} = \frac{REDF^{(ISTAG-1)}}{REDF^{(ISTAG)}}$$

Clearly, $RR^{(ISTAG)}$, $BI^{(ISTAG)}$ and $REDF^{(ISTAG)}$ contain all the information in the original state variables $BLUF^{(ISTAG)}$ and $REDF^{(ISTAG)}$.

Third, let the Red force at the beginning of the compaign be $REDF^{(MSTAGE)}$, then $REDF^{(MSTAGE-1)} = REDF^{(MSTAGE)} \times RR^{(MSTAGE)}$ and in general $REDF^{(ISTAG)}$ can be calculated by $REDF^{(ISTAG)} = REDF^{(MSTAG)} \times RR^{(MSTAG)} \times RR^{(MSTAGE-1)} \times \ldots \times RR^{(ISTAG+1)}$.

Now $BI^{(ISTAG)}$ and $RR^{(ISTAG)}$ together with $REDF^{(MSTAGE)}$ contain all of the information in the original state variables $BLUF^{(ISTAG)}$ and $REDF^{(ISTAG)}$.

Finally note that $BI^{(ISTAG-1)}$ is a function of only $BI^{(ISTAG)}$ and the strategies chosen in stage ISTAG (Equation (7-9))

$$\begin{aligned} BI^{(ISTAG-1)} &= \frac{BLUF^{(ISTAG-1)}}{REDF^{(ISTAG-1)}} \\ &= \{\max(BI^{(ISTAG)} \times FBNC,\ BI^{(ISTAG)} - BABA \\ &\quad \times \max[0, s(K,2) - BAD \times BI^{(ISTAG)} \times s(J,3)])\} \\ &\quad / \{\max(FRNC,\ 1 - BABA \times \max[0, BI^{(ISTAG)} \\ &\quad \times s(J,2) - RAD \times s(K,3)])\} \end{aligned} \qquad (7\text{-}9)$$

In addition, note that $RR^{(ISTAG)}$ is a function of only $BI^{(ISTAG)}$ and the strategies (Equation (7-10))

$$RR^{(ISTAG)} = \frac{REDF^{(ISTAG-1)}}{REDF^{(ISTAG)}}$$

$$= \max(FRNC, 1-BABA \times \max[0, BI^{(ISTAG)} \times s(J,2) - RAD \times s(K,3)]) \qquad (7\text{-}10)$$

Hence only $BI^{(ISTAG)}$ and the initial Red force, $REDF^{(MSTAGE)}$ need be maintained to determine the state transition. If the payoff function can be expressed as only a function of $BI^{(ISTAG)}$ then we have reduced the two-state variable system to an equivalent system with one state variable.

The payoff value equation (7-6) is a function of $REDF^{(ISTAG)}$ and the force ratio $BI^{(ISTAG)}$. Let $T(BI^{(ISTAG)},J,K)$ be the payoff value assuming $REDF^{(ISTAG)}$ equals one and Blue plays J and Red plays K at stage ISTAG.

$$T(BI^{(ISTAG)},J,K) = BCAS \times BI^{(ISTAG)} \times s(J,1) - RCAS \times s(K,1) \qquad (7\text{-}11)$$

Thus the payoff with an arbitrary Red force of $REDF^{(ISTAG)}$ is

$$PAYOFF^{(ISTAG)} = REDF^{(ISTAG)} \times T(BI^{(ISTAG)},J,K)$$

$$= REDF^{(MSTAGE)} \times RR^{(MSTAGE)} \times RR^{(MSTAGE-1)} \times \ldots \times RR^{(ISTAG-1)} \times T(BI^{(ISTAG)},J,K) \qquad (7\text{-}12)$$

Thus the $\text{PAYOFF}^{(n)}$ can be calculated knowing only $\text{REDF}^{(\text{MSTAGE})}$ and $\text{BI}^{(\text{ISTAG})}$ for all values of ISTAG between MSTAGE and n. For the remainder of this chapter we will use the single state variable $\text{BI}^{(\text{ISTAG})}$. This reduces the size of the problem by the square root of the number of force sizes that could be considered in the two-state variable model.

In terms of the single state variable model, Blue wants to maximize (see Equation (5-5))

$$\begin{aligned} M = {} & \text{REDF}^{(\text{MSTAGE})}[T(\text{BI}^{(\text{MSTAGE})},J,K) + \text{RR}^{(\text{MSTAGE})} \\ & [T(\text{BI}^{(\text{MSTAGE-1})},J,K) + \text{RR}^{(\text{MSTAGE-1})} \\ & [T(\text{BI}^{(\text{MSTAGE-2})},J,K) + \text{RR}^{(\text{MSTAGE-2})} \\ & [\ldots + \text{RR}^{(3)}[T(\text{BI}^{(2)},J,K) + \text{RR}^{(2)} \\ & [T(\text{BI}^{(1)},J,K)]]] \quad \ldots]] \end{aligned} \tag{7-13}$$

subject to Equations (7-9) and (7-10).

The dynamic programming approach discussed in the next section will be used to obtain a numerical solution for this problem.

Dynamic Programming Approach. An overview of dynamic programming appears in Appendix B. Dynamic programming is a recursive algorithm that normally begins with an analysis of the last stage.

At the last engagement of the campaign (ISTAG=1) the Blue wants to play a strategy J to maximin the game value ($T(BI^{(ISTAG)},J,K)$), while Red wants to select a strategy K that will minimax the game value. Let $Y(BI^{(ISTAG)})$ denote the optimal game value for stage ISTAG assuming $REDF^{(ISTAG)} = 1$ and entering force ratio $BI^{(ISTAG)}$. $Y(BI^{(1)})$ can be found using the linear programming algorithm to solve the one-stage matrix $T(BI^{(ISTAG)})$ (Ref Chapter II). By changing the value of $BI^{(1)}$ to scan (theoretically) all the possible values of the force ratio, one can get the corresponding last stage (ISTAG=1) optimal game value $Y(BI^{(1)})$ for all values of $BI^{(1)}$.

Suppose we are at the next to last stage (ISTAG=2). Both Blue and Red know that this is not the last engagement. Blue wants to maximize the sum of the game value for this stage and for the next (last) stage while Red wants to minimize them. If Blue plays his Jth strategy and Red plays his Kth strategy, then the payoff of this stage will be $T(BI^{(2)},J,K)$ multiplied by the REDF at the beginning of this stage. As a result of playing those strategies the new force ratio will be $BI^{(1)}$ and the new Red ratio will be $RR^{(2)}$. As the optimal game value of ISTAG=1 is available ($Y(BI^{(1)})$) for each possible entering force ratio, $T,BI^{(1)}$, then one can find the payoff from playing J and K at ISTAG=2 and then optimally at ISTAG=1.

In calculating $Y(BI^{(1)})$ at ISTAG=1 it was assumed that the Red has a unit force but as the Red force is reduced to the ratio as a result of playing the second stage, then the corresponding value will be $Y(BI^{(1)}) \times RR^{(2)}$. The value $Y(BI^{(1)}) \times RR^{(2)} + T(BI^{(2)},J,K)$ represents the payoff if, at the start of the next to last stage (ISTAG=2), we have force ratio BI and REDF=1 and Blue plays J, Red plays K (which lead to a new force ratio $BI^{(1)}$ and a new Red ratio $RR^{(2)}$ and then the optimal strategies are played at the last stage.

Repeating this calculation for each combination of J,K one can get the payoff matrix of the second stage. Linear programming can be used to obtain the optimal game value of this stage $Y(BI^{(2)})$. By varying $BI^{(2)}$ over its full range of potential values we can obtain the corresponding game value $Y(BI^{(2)})$ for all values of $BI^{(2)}$. Next we move to the next stage (ISTAG=3) and repeat the process.

Solution Approximation. As mentioned earlier, it is impossible to find the optimal values of the game $(Y(BI)^{(ISTAG)})$ at each stage for all possible values of the force ratio $(BI^{(ISTAG)})$. Instead, the user specifies the force ratios at which exact solution of the game will be calculated. These values of the force ratio are called grid points. Then at each stage the optimal value of the game $Y(BI)^{(ISTAG)}$ is calculated for each grid point and a cubic spline interpolation is used to estimate the optimal game value for all force ratios that do not exactly match a grid point (see Appendix A).

VIII. Verification and Validation

In the development of a computerized model, two of the most important stages the builder must accomplish are verification and validation. Without them the model formulations, preparation, and translation into an acceptable computer language are meaningless. This chapter will present the procedures used to verify and validate the linear model.

Differentiation between verification and validation is difficult since they are not independent processes. However, verification is generally viewed as insuring that the model behaves the way it was designed. Validation consists of testing the agreement between the behavior of the model and the real system (Ref 11:30).

Verification Tests

The following verification tests were used to demonstrate that there are no logical or computational errors in the computer program (Ref 5:119). The first test of the model was to demand that its behavior not be obviously implausible. In the early development of the model, the implausible results are apt to be of a gross nature. For example, in a tactical situation, the model may indicate a negative force ratio in the sense that one side has a negative number of aircraft. These errors were quite easy to detect and rectify.

Another effective test is to attempt to check model behavior using extreme levels of flow in the system. Model behavior is more unpredictable under normal operating conditions. For example, in the linear model results were observed at a force ratio = 0, when the Blue has no force while the Red has it full force $REDF^{(ISTAG)}$. In this case Red should assign all of his force at all stages to close air support. Another test was made where both sides have the same force size with the same capability. The expected result is a game value of zero for all the stages.

Once these obvious checks were made, attention was directed at more representative performance. The multiple node test considered whether or not the model would provide different behavior when presented with different inputs. In applying this test on the model, the Red and Blue sides were given equivalent capabilities fixed at one (i.e., each Red defender prevents exactly one Blue attacker from reaching the air base). Table VIII-1 gives the optimal game value $Y(BI^{(ISTAG)})$ for ISTAG from one to ten and $BI^{(ISTAG)}$ from zero to twelve. Recall that the value $Y(BI^{(ISTAG)})$ is the optimal game value at force ratio $BI^{(ISTAG)}$ at stage ISTAG assuming $REDF^{(ISTAG)} = 1$. To calculate the actual game value we multiply $Y(I^{(ISTAG)})$ by

TABLE VIII-1

THE LINEAR ALGORITHM OPTIMAL GAME VALUE OF M=3

FR	STG1	STG2	STG3	STG4	STG5	STG6	STG7	STG8	STG9	STG10
0.00	-1.00	-2.00	-3.00	-4.00	-5.00	-6.00	-7.00	-8.00	-9.00	-10.00
.05	-.95	-1.90	-2.80	-3.65	-4.47	-5.42	-6.37	-7.33	-8.32	-9.32
.10	-.90	-1.80	-2.60	-3.37	-4.32	-5.30	-6.29	-7.28	-8.27	-9.27
.15	-.85	-1.70	-2.40	-3.29	-4.27	-5.25	-6.24	-7.23	-8.22	-9.22
.20	-.80	-1.60	-2.30	-3.22	-4.06	-4.98	-5.96	-6.94	-7.93	-8.93
.25	-.75	-1.50	-2.22	-2.99	-3.93	-4.90	-5.89	-6.88	-7.87	-8.86
.30	-.70	-1.40	-2.10	-2.91	-3.86	-4.82	-5.81	-6.80	-7.79	-8.78
.40	-.60	-1.20	-1.80	-2.58	-3.45	-4.41	-5.37	-6.34	-7.32	-8.30
.50	-.50	-1.00	-1.50	-2.25	-3.07	-3.93	-4.82	-5.74	-6.67	-7.57
.75	-.25	-.50	-.75	-1.12	-1.63	-2.22	-2.86	-3.53	-4.23	-4.94
1.00	0.00	0.00	0.00	.00	.00	.00	0.00	.00	0.00	.00
1.25	.25	.50	.75	1.13	1.65	2.27	2.97	3.71	4.49	5.29
1.50	.50	1.00	1.50	2.25	3.19	4.24	5.36	6.52	7.70	8.90
1.75	.75	1.50	2.25	3.33	4.62	6.01	7.47	8.97	10.51	12.02
2.00	1.00	2.00	3.00	4.50	6.16	7.91	9.73	11.59	13.40	15.21
2.50	1.50	3.00	4.50	6.46	8.64	11.00	13.39	15.81	18.24	20.68
3.00	2.00	4.00	6.00	8.58	11.42	14.36	17.32	20.30	23.28	26.26
3.50	2.50	5.00	7.47	10.28	13.53	16.91	20.37	23.84	27.31	30.79
4.00	3.00	6.00	8.89	11.94	15.70	19.46	23.41	27.37	31.35	35.33
4.50	3.50	7.00	10.25	14.00	17.88	22.16	26.60	31.06	35.53	40.00
5.00	4.00	8.00	11.50	16.07	20.11	24.85	29.78	34.73	39.69	44.67
5.50	4.50	9.00	12.75	17.94	22.57	27.41	32.83	38.28	43.74	49.21
6.00	5.00	10.00	14.00	19.67	25.50	31.40	37.33	43.29	49.25	55.22
6.50	5.50	11.00	15.50	21.39	27.71	34.10	40.53	46.98	53.44	59.91
7.00	6.00	12.00	17.00	23.11	29.92	36.80	43.72	50.67	57.63	64.59
7.50	6.50	13.00	18.50	24.83	32.13	39.50	46.92	54.36	61.81	69.28
8.00	7.00	14.00	20.00	26.56	34.33	42.20	50.11	58.05	66.00	73.96
8.50	7.50	15.00	21.50	28.28	36.54	44.90	53.31	61.74	70.19	78.65
9.00	8.00	16.00	23.00	30.00	38.75	47.60	56.50	65.43	74.38	83.33
9.50	8.50	17.00	24.50	31.83	40.96	50.30	59.69	69.12	78.56	88.02
10.00	9.00	18.00	26.00	33.67	43.17	53.00	62.89	72.81	82.75	92.70
10.50	9.50	19.00	27.50	35.50	45.38	55.70	66.08	76.50	86.94	97.39
11.00	10.00	20.00	29.00	37.33	47.58	58.40	69.28	80.19	91.13	102.07
11.50	10.50	21.00	30.50	39.17	49.79	61.10	72.47	83.88	95.31	106.76
12.00	11.00	22.00	32.00	41.00	52.00	63.80	75.67	87.57	99.50	111.44

$REDF^{(ISTAG)}$. For example, if $REDF^{(3)} = 100$ and $BLUF^{(3)} = 50$ the force ratio $BI^{(3)}$ will be $BLUF^{(3)}/REDF^{(3)}$, or .5. For a three-stage game with $BI^{(3)} = .5$ the linear algorithm found $Y(.5)^{(3)} = -1.5$. That is to say, the difference between the total number of Blue aircraft and the total number of Red aircraft sent on close air support through the three stages equals $REDF^{(3)} \times Y_3(.5)$ or -150. The same results with a positive sign should occur if $REDF^{(3)} = 50$ and $BLUF^{(3)} = 100$ for three stages (as the aircraft for both sides are assumed to have the same capabilities). In this case $BI^{(3)} = 2$. The algorithm found $Y(2^{(3)}) = 3.0$. Hence the payoff to Blue for the three-stage campaign was 50 x 3.0 = 150, as expected. Similar tests were conducted using force ratio pairs of {.1, 10}, {.2, 5}, {.24, 4}, {.4, 2.5} and {.5, 2} for air wars lasting from one to ten stages. Some small errors were expected due to the cubic spline interpolation approximation and the linear extrapolation beyond the last grid point. A percentage error is calculated for each of these tests using

$$\% \text{ Error} = \frac{Y(BI^{(ISTAG)}) - (-Y(1/BI^{(ISTAG)})/BI^{(ISTAG)})}{Y(BI^{(ISTAG)})} \times 100$$

Table VIII-2 shows the results of these tests. The first line for each case is $Y(BI^{(ISTAG)})$. The second line is $Y(1/BI^{(ISTAG)})$. The third line is the percentage error.

TABLE VIII-2

THE PERCENTAGE ERROR IN THE LINEAR ALGORITHM

ERROR	FR	STG3	STG4	STG5	STG6	STG7	STG8	STG9	STG10
	.10	-2.60	-3.37	-4.32	-5.30	-6.29	-7.28	-8.27	-9.27
	10.00	26.00	33.67	43.17	53.00	62.89	72.81	82.75	92.70
%		0.000	.000	.000	.000	.000	.000	.000	.000
	.20	-2.30	-3.22	-4.06	-4.98	-5.96	-6.94	-7.93	-8.93
	5.00	11.50	16.07	20.11	24.85	29.78	34.73	39.69	44.67
%		.000	-.027	-1.016	-.123	-.027	.034	.062	.054
	.25	-2.22	-2.99	-3.93	-4.90	-5.89	-6.88	-7.87	-8.86
	4.00	8.89	11.94	15.70	19.46	23.41	27.37	31.35	35.33
%		.000	-.150	-.208	-.751	-.584	-.473	-.398	-.350
	.40	-1.80	-2.58	-3.45	-4.41	-5.37	-6.34	-7.32	-8.30
	2.50	4.50	6.46	8.64	11.00	13.39	15.81	18.24	20.68
%		.000	.152	.105	-.166	-.292	-.306	-.308	-.289
	.50	-1.50	-2.25	-3.07	-3.93	-4.82	-5.74	-6.67	-7.57
	2.00	3.00	4.50	6.16	7.91	9.73	11.59	13.40	15.21
%		.000	.002	.390	.727	.908	.895	.475	.508

There was no error in the first three stages. The maximum error found was just over one percent.

Validation Test

To validate a war game, a means of building confidence in the game's ability to achieve its objectives must be devised. An important distinction between verification and validation is that models can be completely verified, while complete validation is impossible. Richard L. Van Horn (Ref 13:247) suggests that a model may be considered valid when it has achieved an acceptable level of confidence.

Berkovitz and Dresher (3:155) have published results of a linear game when the individual weapon systems on each side have equal capabilities and the fraction of the opponent's force caught on the ground in an air base attack is one. Although they did simplify the problem with the above assumptions, they were also able to obtain solutions for a continuum of strategies. Table VIII-3 is a copy of their results for up to an eight-stage game. That is, Berkovitz and Dresher do not require that the fractions of the Blue force sent on CAS, ABA or AD be 0, 1/3, 2/3, or I (M=3). Instead, they allow any fraction of the force to be sent on CAS, ABA or AD (of course they require the individual fractions to sum to one). Because of their "continuous" feature of the Berkovitz-Dresher results, some differences between the two results should exist.

TABLE VIII-3

A COPY OF BERKOVITZ-DRESHER RESULTS

OPTIMAL ALLOCATION OF FORCES AMONG THREE TASKS
(Strong side having p forces and weak side having q forces, $p > q$)

Duration of campaign (no. of strikes remaining in campaign)	*Relative initial strengths of opponents (ratio of strong side to weak side) p/q*	*Optimal initial allocation by strong side (force size allocated to)*			*Optimal initial allocation by weak side (probability of concentrating forces on)*			*Value of game*
		Counter Air	Air Defense	Ground Support	Counter Air	Air Defense	Ground Support	
1	1.00 to ∞	0	0	p	0	0	1	$p - q$
2	1.00 to ∞	0	0	p	0	0	1	$2(p - q)$
3	1.00 to 2.00	q	0	$p - q$	0.50	0.50	0	$3(p - q)$
	2.00 to ∞	$1.5q$	$0.5q$	$p - 2q$	0.50	0.50	0	$3(p - q)$
4	1.00 to 2.33	$0.5p + 0.5q$	$0.5p - 0.5q$	0	0.50	0.50	0	$4.5(p - q)$
	2.33 to ∞	$1.67q$	$0.67q$	$p - 2.33$	0.33	0.33	0.33	$4.00p - 3.33q$
5	1.00 to 1.7	$0.41p + 0.59q$	$0.59p - 0.59q$	0	0.53	0.47	0	$6.35p - 6.35q$
	1.70 to 2.45	$0.55p + 0.36q$	$0.45p - 0.36q$	0	0.45	0.55	0	$5.82p - 5.45q$
	2.45 to ∞	$1.70q$	$0.75q$	$p - 2.45$	0.25	0.30	0.45	$5.00p - 3.45q$
6	1.00 to 1.44	$0.32p + 0.68q$	$0.68p - 0.68q$	0	0.56	0.44	0	$8.39p - 8.39q$
	1.44 to 1.78	$0.40p + 0.56q$	$0.60p - 0.56q$	0	0.52	0.48	0	$8.00p - 7.83q$
	1.78 to 2.51	$0.59p + 0.22q$	$0.41p - 0.22q$	0	0.41	0.59	0	$7.04p - 6.12q$
	2.51 to ∞	$1.71q$	$0.80q$	$p - 2.51$	0.20	0.29	0.51	$6.00p - 3.51q$
7	1.00 to 1.29	$0.25p + 0.75q$	$0.75p - 0.75q$	0	0.58	0.42	0	$10.50p - 10.50q$
	1.29 to 1.53	$0.29p + 0.70q$	$0.71p - 0.70q$	0	0.57	0.43	0	$10.26p - 10.19q$
	1.53 to 1.84	$0.41p + 0.51q$	$0.59p - 0.51q$	0	0.50	0.50	0	$9.54p - 9.09q$
	1.84 to 2.55	$0.63p + 0.11q$	$0.37p - 0.11q$	0	0.37	0.63	0	$8.21p - 6.61q$
	2.55 to ∞	$1.72q$	$0.81q$	$p - 2.55$	0.17	0.28	0.55	$7.00p - 3.55q$
8	1.00 to 1.25	$0.20p + 0.80q$	$0.80p - 0.80q$	0	0.60	0.40	0	$12.60p - 12.60q$
	1.25 to 1.40	$0.22p + 0.78q$	$0.78p - 0.78q$	0	0.59	0.41	0	$12.48p - 12.45q$
	1.40 to 1.59	$0.28p + 0.69q$	$0.72p - 0.69q$	0	0.56	0.44	0	$12.06p - 11.86q$
	1.59 to 1.88	$0.42p + 0.46q$	$0.58p - 0.46q$	0	0.49	0.51	0	$11.03p - 10.22q$
	1.88 to 2.58	$0.66p + 0.01q$	$0.34p - 0.01q$	0	0.31	0.66	0	$9.36p - 7.07q$
	2.58 to ∞	$1.72q$	$0.86q$	$p - 2.58$	0.14	0.28	0.58	$8.00p - 3.58q$

For example, consider the case of a three-stage game with $REDF^{(3)} = 100$ and $BLUF^{(3)} = 5$. In the continuous case the optimal strategy for the third stage (start of the war) is that Red sent 5 aircraft in air base attack and 95 aircraft in close air support, while the optimal strategy for Blue is to send his force on air base attack and/or air defense. The possible results of this stage is a payoff equal to -95 for Blue with the remaining force sizes (Table VIII-4) depending on the Blue strategy.

TABLE VIII-4

THE RESULTS OF CONTINUOUS STRATEGIES

Blue Strategy			
ABA	AD	$REDF^{(2)}$	$BLUF^{(2)}$
5	0	95	0
4	1	96	1
3	2	97	2
2	3	98	3
1	4	99	4
0	5	100	5

In the second and first stage, one of the optimal strategies for both sides is to send all the rest of their forces on close air support missions. The total payoff from Berkovitz and Dresher is $-95 + 2\ (BLUF^{(2)} - REDF^{(2)}) = -285$. This result is given in the Berkovitz-Dresher Table (Table VIII-3).

Consider now the same problem but with a finite set of strategies, i.e., the fraction of forces sent is an integer multiple of M. Assume M=3, then Red cannot send 5 aircraft on air base attack. Either he sends 33 aircraft on this mission or he send nothing. The obvious solution in this case is that he sends nothing on air base attack and sends all his forces in close air support for all three stages. In the mean while, Blue will send his five planes on air base attack at the third stage (start of war) and send them on close air support missions at the second and first stages. The results are in Table VIII-5.

TABLE VIII-5

THE RESULTS OF DISCRETE STRATEGIES (M=3)

Stage	Blue Strategies CAS	ABA	AD	Red CAS	BLUF$^{(ISTAG)}$	REDF$^{(ISTAG)}$	GV
3	0	5	0	100	5	95	-100
2	0	5	0	95	5	95	- 90
1	5	0	0	95	5	95	- 90
							-280

Thus the optimal payoff obtained using the linear algorithm is -280. This example shows the reason that differences will exist between the results of this special case of the linear algorithm and the Berkovitz-Dresher results. Let $B(BI^{(MSTAG)})$ be the total game value in the Berkovitz-Dresher solution at force ratio BI and total number of stages MSTAGE assuming that Red has initial force of one unit. Table VIII-6 summarizes the Berkovitz-Dresher results for all of the force ratios (grid points) used in the linear algorithm. Table VIII-7 shows the fractional differences between the linear algorithm results (Table VIII-1) and Berkovitz-Dresher results (Table VIII-6) calculated as follows:

$$\text{Fractional Difference} = \frac{Y(BI^{(MSTAG)}) - B(BI^{(MSTAG)})}{Y(BI^{(MSTAG)})}$$

for MSTAGE = 1,2,...,8 and M=3.

Table VIII-8 shows the optimal game value at M=5 and Table VIII-9 shows the fractional difference between the linear algorithm (Table VIII-8) and Berkovitz-Dresher results (Table VIII-6). It was expected that the linear algorithm results for M=5 would universally be closer to the Berkovitz-Dresher results than those obtained for M=3. This proved not to be the case at each grid point. There appears to be an error in the Berkovitz-Dresher table. Another author (Ref 9:76) also found this error.

Comparisons to OPTSA

Until now we have only tried validating the model when aircraft capabilities on both sides are identical. To make sure that the model gives reasonable results when the sides have different capabilities,

TABLE VIII-6

BERKOVITZ AND DRESHER RESULTS AT THE SAME FORCE RATIO USED IN LINEAR ALGORITHM

FR	STG1	STG2	STG3	STG4	STG5	STG6	STG7	STG8
0.00	-1.00	-2.00	-3.00	-4.00	-5.00	-6.00	-7.00	-8.00
.05	-.95	-1.90	-2.85	-3.83	-4.83	-5.82	-6.82	-7.82
.10	-.90	-1.80	-2.70	-3.67	-4.66	-5.65	-6.65	-7.64
.15	-.85	-1.70	-2.55	-3.50	-4.48	-5.47	-6.47	-7.46
.20	-.80	-1.60	-2.40	-3.33	-4.31	-5.30	-6.29	-7.28
.25	-.75	-1.50	-2.25	-3.17	-4.14	-5.12	-6.11	-7.11
.30	-.70	-1.40	-2.10	-3.00	-3.97	-4.95	-5.94	-6.93
.40	-.60	-1.20	-1.80	-2.67	-3.62	-4.59	-5.55	-6.53
.50	-.50	-1.00	-1.50	-2.25	-3.10	-3.98	-4.89	-5.83
.75	-.25	-.50	-.75	-1.13	-1.59	-2.10	-2.62	-3.14
1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
1.25	.25	.50	.75	1.13	1.59	2.10	2.63	3.15
1.50	.50	1.00	1.50	2.25	3.18	4.17	5.20	6.23
1.75	.75	1.50	2.25	3.38	4.74	6.17	7.61	9.08
2.00	1.00	2.00	3.00	4.50	6.19	7.96	9.78	11.65
2.50	1.50	3.00	4.50	6.67	9.05	11.48	13.89	16.33
3.00	2.00	4.00	6.00	8.67	11.55	14.49	17.45	20.42
3.50	2.50	5.00	7.50	10.67	14.05	17.49	20.95	24.42
4.00	3.00	6.00	9.00	12.67	16.55	20.49	24.45	28.42
4.50	3.50	7.00	10.50	14.67	19.05	23.49	27.95	32.42
5.00	4.00	8.00	12.00	16.67	21.55	26.49	31.45	36.42
5.50	4.50	9.00	13.50	18.67	24.05	29.49	34.95	40.42
6.00	5.00	10.00	15.00	20.67	26.55	32.49	38.45	44.42
6.50	5.50	11.00	16.50	22.67	29.05	35.49	41.95	48.42
7.00	6.00	12.00	18.00	24.67	31.55	38.49	45.45	52.42
7.50	6.50	13.00	19.50	26.67	34.05	41.49	48.95	56.42
8.00	7.00	14.00	21.00	28.67	36.55	44.49	52.45	60.42
8.50	7.50	15.00	22.50	30.67	39.05	47.49	55.95	64.42
9.00	8.00	16.00	24.00	32.67	41.55	50.49	59.45	68.42
9.50	8.50	17.00	25.50	34.67	44.05	53.49	62.95	72.42
10.00	9.00	18.00	27.00	36.67	46.55	56.49	66.45	76.42
10.50	9.50	19.00	28.50	38.67	49.05	59.49	69.95	80.42
11.00	10.00	20.00	30.00	40.67	51.55	62.49	73.45	84.42
11.50	10.50	21.00	31.50	42.67	54.05	65.49	76.95	88.42
12.00	11.00	22.00	33.00	44.67	56.55	68.49	80.45	92.42

TABLE VIII-7

THE FRACTIONAL DIFFERENCE BETWEEN THE LINEAR ALGORITHM RESULTS AT M=3 AND BERKOVITZ-DRESHER RESULTS

FR	STG1	STG2	STG3	STG4	STG5	STG6	STG7	STG8
0.00	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
.05	0.000	0.000	-.018	-.050	-.081	-.075	-.072	-.067
.10	0.000	0.000	-.038	-.089	-.078	-.066	-.057	-.050
.15	0.000	0.000	-.062	-.063	-.051	-.043	-.037	-.032
.20	0.000	0.000	-.043	-.037	-.061	-.065	-.056	-.049
.25	0.000	0.000	-.013	-.059	-.052	-.045	-.038	-.033
.30	0.000	0.000	-.002	-.030	-.028	-.026	-.022	-.019
.40	0.000	0.000	-.000	-.035	-.049	-.042	-.034	-.030
.50	0.000	0.000	-.000	-.000	-.009	-.013	-.014	-.014
.75	0.000	0.000	0.000	-.000	.029	.055	.084	.110
1.00								
1.25	0.000	0.000	0.000	.000	.036	.077	.115	.151
1.50	0.000	0.000	0.000	0.000	.005	.018	.030	.044
1.75	0.000	0.000	0.000	-.014	-.025	-.026	-.018	-.012
2.00	0.000	0.000	0.000	0.000	-.005	-.006	-.005	-.005
2.50	0.000	0.000	0.000	-.033	-.048	-.044	-.037	-.033
3.00	0.000	0.000	0.000	-.010	-.012	-.009	-.007	-.006
3.50	0.000	0.000	-.004	-.038	-.039	-.034	-.029	-.025
4.00	0.000	0.000	-.012	-.061	-.054	-.053	-.045	-.038
4.50	0.000	0.000	-.024	-.048	-.066	-.060	-.051	-.044
5.00	0.000	0.000	-.043	-.037	-.072	-.066	-.056	-.049
5.50	0.000	0.000	-.059	-.040	-.066	-.076	-.064	-.056
6.00	0.000	0.000	-.071	-.051	-.041	-.035	-.030	-.026
6.50	0.000	0.000	-.065	-.060	-.048	-.041	-.035	-.031
7.00	0.000	0.000	-.059	-.067	-.055	-.046	-.040	-.035
7.50	0.000	0.000	-.054	-.074	-.060	-.050	-.043	-.038
8.00	0.000	0.000	-.050	-.080	-.065	-.054	-.047	-.041
8.50	0.000	0.000	-.047	-.085	-.069	-.058	-.050	-.043
9.00	0.000	0.000	-.043	-.089	-.072	-.061	-.052	-.046
9.50	0.000	0.000	-.041	-.089	-.075	-.063	-.055	-.048
10.00	0.000	0.000	-.038	-.089	-.078	-.066	-.057	-.050
10.50	0.000	0.000	-.036	-.089	-.081	-.068	-.059	-.051
11.00	0.000	0.000	-.034	-.089	-.083	-.070	-.060	-.053
11.50	0.000	0.000	-.033	-.089	-.086	-.072	-.062	-.054
12.00	0.000	0.000	-.031	-.090	-.088	-.074	-.063	-.055

TABLE VIII-8

THE LINEAR ALGORITHM OPTIMAL GAME VALUES AT M=5

FR	STG1	STG2	STG3	STG4	STG5	STG6	STG7	STG8	STG9	STG10
0.00	-1.00	-2.00	-3.00	-4.00	-5.00	-6.00	-7.00	-8.00	-9.00	-10.00
.05	-.95	-1.90	-2.80	-3.65	-4.60	-5.59	-6.58	-7.58	-8.57	-9.57
.10	-.90	-1.80	-2.60	-3.57	-4.55	-5.54	-6.53	-7.53	-8.52	-9.52
.15	-.85	-1.70	-2.53	-3.39	-4.36	-5.35	-6.34	-7.33	-8.33	-9.33
.20	-.80	-1.60	-2.40	-3.31	-4.28	-5.26	-6.26	-7.25	-8.25	-9.25
.25	-.75	-1.50	-2.25	-3.11	-4.08	-5.05	-6.02	-6.99	-7.97	-8.96
.30	-.70	-1.40	-2.10	-3.00	-3.93	-4.90	-5.88	-6.86	-7.85	-8.84
.40	-.60	-1.20	-1.80	-2.67	-3.62	-4.58	-5.54	-6.51	-7.49	-8.47
.50	-.50	-1.00	-1.50	-2.25	-3.09	-3.98	-4.89	-5.80	-6.73	-7.67
.75	-.25	-.50	-.75	-1.13	-1.65	-2.25	-2.89	-3.55	-4.24	-4.94
1.00	0.00	0.00	0.00	-.00	-.00	.00	.00	.00	-.00	-.00
1.25	.25	.50	.75	1.13	1.66	2.30	2.99	3.74	4.51	5.32
1.50	.50	1.00	1.50	2.25	3.20	4.23	5.34	6.48	7.66	8.85
1.75	.75	1.50	2.25	3.38	4.68	6.05	7.49	8.98	10.50	12.04
2.00	1.00	2.00	3.00	4.50	6.18	7.96	9.76	11.59	13.45	15.33
2.50	1.50	3.00	4.50	6.67	9.04	11.42	13.83	16.26	18.71	21.16
3.00	2.00	4.00	6.00	8.58	11.28	14.18	17.12	20.08	23.04	26.02
3.50	2.50	5.00	7.50	10.62	13.89	17.23	20.65	24.09	27.55	31.01
4.00	3.00	6.00	9.00	12.43	16.21	20.14	24.04	27.95	31.85	35.76
4.50	3.50	7.00	10.50	14.59	18.58	23.01	27.46	31.93	36.41	40.89
5.00	4.00	8.00	12.00	16.58	21.36	26.30	31.27	36.24	41.22	46.21
5.50	4.50	9.00	13.49	18.40	23.59	29.03	34.49	39.96	45.44	50.92
6.00	5.00	10.00	14.96	20.20	25.82	31.75	37.71	43.68	49.66	55.64
6.50	5.50	11.00	16.41	22.02	28.15	34.58	41.03	47.50	53.97	60.45
7.00	6.00	12.00	17.84	23.92	30.48	37.40	44.34	51.30	58.28	65.25
7.50	6.50	13.00	19.25	26.00	s2.72	40.13	47.57	55.03	62.50	69.98
8.00	7.00	14.00	20.60	28.05	34.96	42.86	50.80	58.76	66.72	74.70
8.50	7.50	15.00	21.95	30.08	37.29	45.68	54.11	62.57	71.03	79.50
9.00	8.00	16.00	23.30	32.00	39.64	48.50	57.42	66.37	75.33	84.30
9.50	8.50	17.00	24.65	33.83	41.99	51.23	60.66	70.10	79.56	89.03
10.00	9.00	18.00	26.00	35.67	45.50	55.40	65.33	75.29	85.25	95.22
10.50	9.50	19.00	27.50	37.50	47.83	58.22	63.65	79.10	89.56	100.03
11.00	10.00	20.00	29.00	39.33	50.15	61.04	71.97	82.91	93.88	104.84
11.50	10.50	21.00	30.50	41.17	52.48	63.86	75.28	86.73	98.19	109.66
12.00	11.00	22.00	32.00	43.00	54.80	66.68	78.60	90.54	102.50	114.47

TABLE VIII-9

THE DIFFERENCE BETWEEN THE LINEAR ALGORITHM RESULTS AT M=5 AND BERKOVITZ-DRESHER RESULTS

FR	STG1	STG2	STG3	STG4	STG5	STG6	STG7	STG8
0.00	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
.05	0.000	0.000	-.018	-.050	-.049	-.042	-.036	-.032
.10	0.000	0.000	-.038	-.028	-.023	-.020	-.017	-.015
.15	0.000	0.000	-.007	-.032	-.028	-.024	-.020	-.018
.20	0.000	0.000	-.000	-.006	-.007	-.006	-.005	-.004
.25	0.000	0.000	-.000	-.019	-.015	-.014	-.015	-.016
.30	0.000	0.000	-.000	-.000	-.008	-.010	-.010	-.009
.40	0.000	0.000	-.000	-.001	-.001	-.002	-.002	-.003
.50	0.000	0.000	-.000	-.000	-.001	-.000	-.001	-.004
.75	0.000	0.000	-.000	.000	.036	.066	.094	.116
1.00								
1.25	0.000	0.000	0.000	.000	.042	.086	.123	.157
1.50	0.000	0.000	0.000	-.000	.007	.015	.026	.039
1.75	0.000	0.000	0.000	.000	-.012	-.021	-.015	-.011
2.00	0.000	0.000	0.000	0.000	-.001	-.000	-.002	-.005
2.50	0.000	0.000	0.000	-.000	-.001	-.005	-.004	-.004
3.00	0.000	0.000	0.000	-.011	-.024	-.022	-.019	-.017
3.50	0.000	0.000	0.000	-.005	-.011	-.015	-.015	-.014
4.00	0.000	0.000	0.000	-.019	-.021	-.017	-.017	-.017
4.50	0.000	0.000	0.000	-.006	-.025	-.021	-.018	-.015
5.00	0.000	0.000	0.000	-.005	-.009	-.007	-.006	-.005
5.50	0.000	0.000	-.001	-.015	-.020	-.016	-.013	-.011
6.00	0.000	0.000	-.003	-.023	-.028	-.023	-.020	-.017
6.50	0.000	0.000	-.005	-.029	-.032	-.026	-.022	-.019
7.00	0.000	0.000	-.009	-.031	-.035	-.029	-.025	-.022
7.50	0.000	0.000	-.013	-.026	-.041	-.034	-.029	-.025
8.00	0.000	0.000	-.019	-.022	-.045	-.038	-.032	-.028
8.50	0.000	0.000	-.025	-.020	-.047	-.040	-.034	-.030
9.00	0.000	0.000	-.030	-.021	-.048	-.041	-.035	-.031
9.50	0.000	0.000	-.034	-.025	-.049	-.044	-.038	-.033
10.00	0.000	0.000	-.038	-.028	-.023	-.020	-.017	-.015
10.50	0.000	0.000	-.036	-.031	-.026	-.022	-.019	-.017
11.00	0.000	0.000	-.034	-.034	-.028	-.024	-.021	-.018
11.50	0.000	0.000	-.033	-.037	-.030	-.026	-.022	-.020
12.00	0.000	0.000	-.031	-.039	-.032	-.027	-.024	-.021

the author compared the linear model results to those obtained using OPTSA. But, since there are no published tables of OPTSA results, the author generated a computer code (Appendix E) to implement OPTSA. This model provides the optimal game value for a three-stage game. In the test, assume $REDF^{(3)} = 100$ and $BLUF^{(3)} = 125$, and all aircraft capabilities equal one (after 51.16 cp second execution time) OPTSA provides a game value of 75, which is exactly the same as given in Tables VIII-1, VIII-6, and VIII-8. Those tables give $Y(1.25^{(3)}) = .75$; i.e., the optimal game value = $.75 \times REDF^{(3)} = 75$. Now execute OPTSA using the following data: $REDF^{(3)} = 100$, $BLUF^{(3)} = 500$, BABA = 1, RABA = 3, BAD = 1, RAD = 3, BCAS = 2, and RCAS = 1 it gives an optimal game value of 2138.74. The linear algorithm with the same data (Table X-1) gives $Y(5^{(3)}) = 21.19$; i.e., the optimal game value = $21.19 \times REDF^{(3)} = 2119$. The error in the linear algorithm is 0.889 percent.

IX. An Interactive Game Algorithm for Training and Assisting the Commander

Introduction

As most operational commanders of general purpose forces know, the ways such forces are employed can have a great impact on the outcome of a conventional tactical campaign. While it is true that larger and better equipped forces win over inferior forces more often than they lose, history (6:1:28) is replete with examples where superior forces were defeated as a result of the unwise employment of the superior forces, and the wise employment of the inferior forces. Using combat simulations, it is also very simple to demonstrate the defeat of superior forces through a combination of "smart" and "dumb" force employment by the victor and the loser, respectively. Even when a "black and white" win or lose is not the issue, it is clear, both from historical and analytical perspectives, that a wide range of outcomes in conventional tactical campaigns can be expected, depending upon how the forces are used.

In this context a "strategy" is a series of daily decisions on allocating air forces among close air support, air base attack, and air defense missions. It depends each day on the capabilities of the two sides, the stage of the campaign, and the other critical variables.

The program GAME is designed to help train and assist the commander with regard to the allocation of tactical air war resources. The program GAME uses the optimal game values provided by the linear algorithm. For any given scenario, GAME computes the "optimal" strategies for each side. The only difference in the GAME algorithm between the assist and training mode is when the computer prints the optimal strategy. When GAME is used in the assist mode, the computer plays the same side with the commander and it prints both sides' optimal strategies. The commander can then specify strategies other than the optimal strategy and determine their effect on the game value. When GAME is used to train the commander, the computer plays the enemy against the commander and it does not print the enemy's "optimal" strategy until after the commander has entered his strategy. The commander then has the option to see one of his optimal strategies. This optimal strategy will appear before the commander has entered his strategy and the game will continue based on the strategy chosen by the commander, not his optimal strategy.

The Algorithm

GAME uses a copy of the results of the linear algorithm. Before running the GAME program, we must execute the linear algorithm program (F3GAME). The program F3GAME asks the player to specify: (1) the maximum number of stages; (2) the aircraft capabilities for both sides; (3) the number of grid points. The number of grid points is used to determine the values of the state variable (force ratio) at which solutions will be obtained. In solving this game, the initial Red force size is assumed to be 1. After running the program F3GAME, it will write on tape 3 the above data plus; (4) the force ratio at each grid point; and (5) the game value for each stage at each grid point. Then, program GAME will execute. First it reads the data from tape 2, stores the force ratio at each grid point in the vector X with size (40), and stores the game values in a vector Y with size (400). Conceptually, this information can be represented as in Figure IX-1 with up to 40 points for each stage. Second, the program then requires the player to select what kind of game he wants to play, training or assisting. Third, the player enters the total number of states he wants to play (MSTAGE) and the number of aircraft available to each side at the start of the game, $\text{BLUF}^{(\text{MSTAGE})}$), $\text{REDF}^{(\text{MSTAGE})}$).

GAME is able to efficiently determine the optimal strategies and the corresponding game value at any arbitrarily selected stage (ISTAG) of the game by solving a single game. The entering force ratio $\text{BI}^{(\text{ISTAG})}$ is given. Using this force ratio GAME generates the

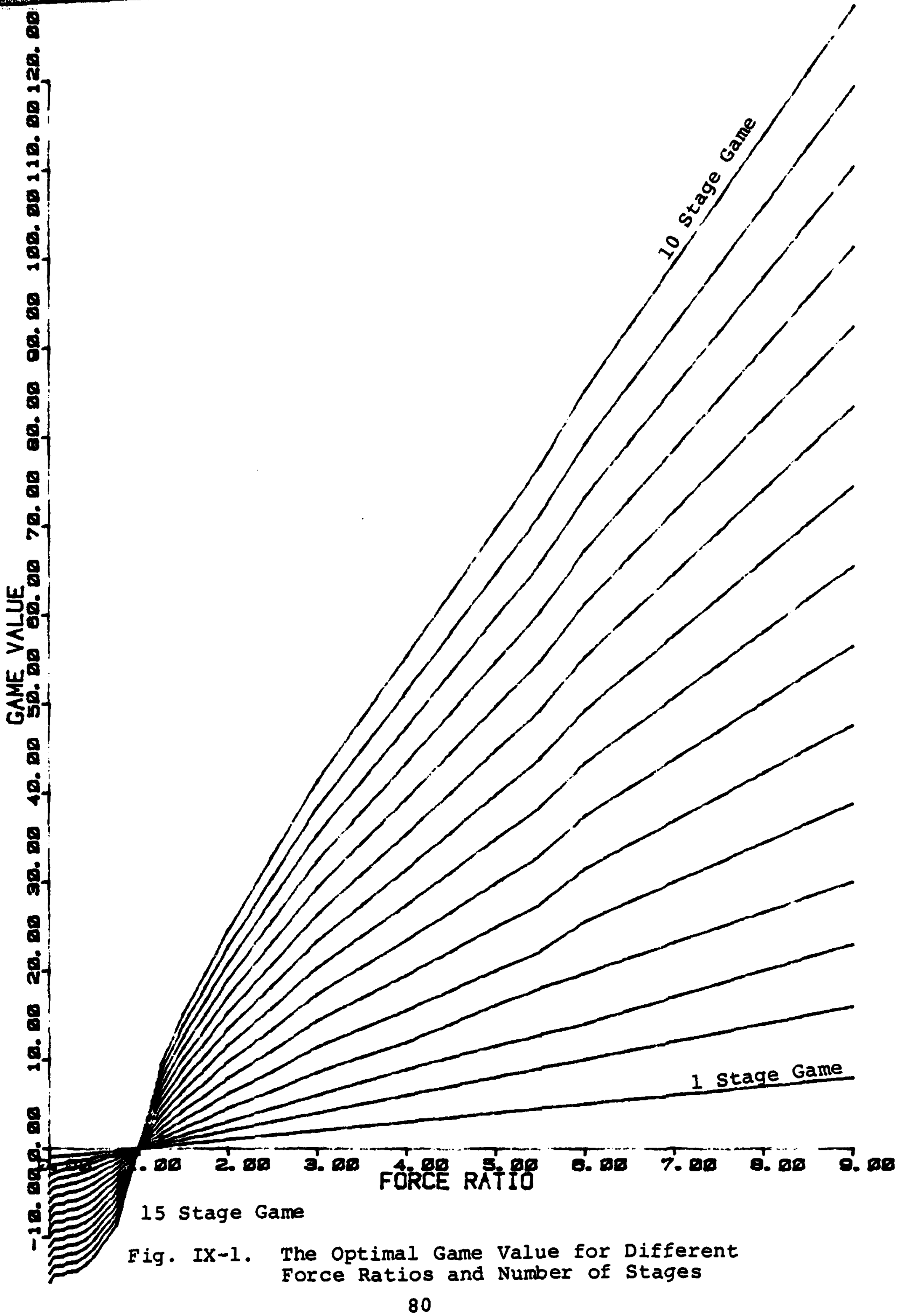


Fig. IX-1. The Optimal Game Value for Different Force Ratios and Number of Stages

one-stage payoff matrix for the game (T(BI$^{(ISTAG)}$,J,K) for J=1,...B, B=1,...R. BI$^{(ISTAG)}$ is also used to generate, using the state transition functions, BI$^{(ISTAG-1)}$ and RR$^{(ISTAG)}$ for any selection of strategies J and K at ISTAG. This resulting force ratio is then used in Figure IX-1, with cubic spline interpolation between grid points, to generate Y(I$^{(ISTAG-1)}$) for all J=1,...B and K=1...R. The sum of the one-stage (T(BI$^{(ISTAG)}$,J,K) and Y(I$^{(ISTAG-1)}$) x RR$^{(ISTAG)}$ values are the payoff values in the single stage game to be optimized. This process is summarized in Figure IX-2. The solution of this game provides the game value and optimal strategies for Blue and Red at ISTAG.

Note that in the training mode, regardless of the strategy selected by the commander, an updated force ratio and game value can be calculated. At the next stage we use the entering force ratio and the procedure discussed above to determine the optimal strategies and game value by again only solving <u>one game</u>!

In the training mode GAME generates a random number and selects, according to this number, one of mixed strategies for the side it is playing. When the commander's strategy is entered into the computer, GAME calculates and prints the game value of this stage, the total game value from his play and the previous play(s), and the new state of the game according to the chosen strategies of both sides. This procedure continues until the end of the game.

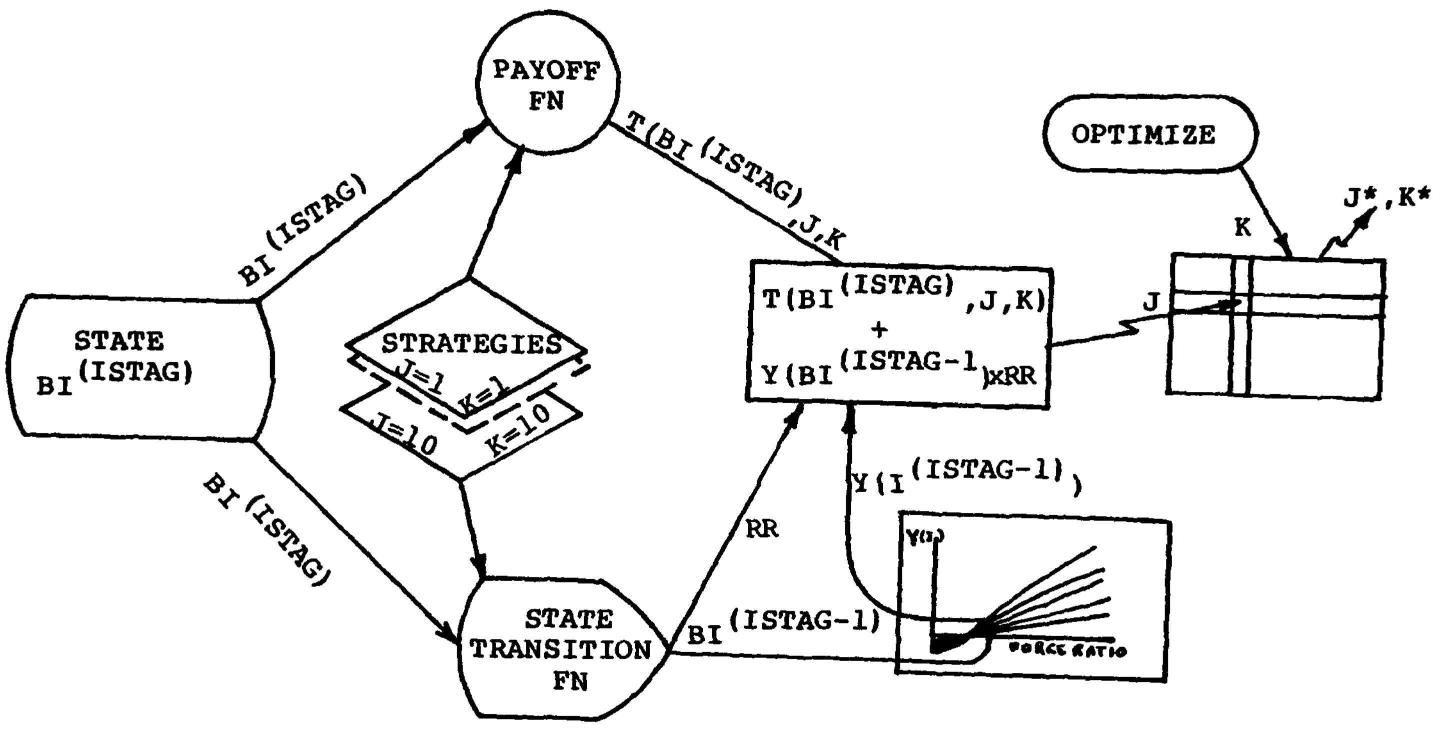


Fig. IX-2. GAME Process to Determine the Optimal Strategies

Recall that the computer strategies are printed before the player's strategy enters the computer in the assist mode and is printed after it enters in the training mode. Appendix C contains the player guide and the code for GAME.

X. Comparison of Weapon Systems

Developing countries are frequently involved in difficult weapon system procurement decisions. For example, suppose a country needs to improve its tactical air force. It may have several different alternative aircrafts available to purchase (e.g., F-15, Mirage, etc.), each with different capabilities and different costs. Given a limited budget, which aircraft should be purchased? This depends on how well each of the alternative aircraft perform against the corresponding forces of a potential enemy. It may also depend on the length of the war.

The purpose of this chapter is to show how multi-stage game theory may be applied to help supply insight to this procurement question. In the next section the importance of using optimal strategies in these force-on-force comparisons is discussed. This is followed by a simplified illustrative example.

The Importance of Optimal Strategies

Many simulations of force-on-force battles begin by stating the enemy strategy (e.g., 15% of the aircraft will be allocated to air defense, 40% to air base attack, and 45% to ground support). At the worst, the allocations are "thought up" in order to have something to make the program run. At best, the allocations are considered a "best estimate" of the enemy's behavior. Typically, a similar procedure is used to develop the strategy to be followed by the friendly forces. These strategies are then "fixed" or held constant throughout the evaluation of several general-purpose forces (Ref 6:29).

Even when the strategies chosen are given "careful thought" and analysts choose "good strategies," a procedure that uses the same strategy in comparing alternative forces makes invalid comparisons. To see why this is so, consider a Blue force that is to be compared against a Red threat. "Good" strategies are selected for both sides, and a campaign is fought. Let the outcome, in terms of some measure of merit, be called V_1. Suppose now that a new weapon system is added to the Blue force and a campaign is fought again, with the same strategies as before, and a more favorable outcome, V_2, is achieved. The statement usually made at this point is, "The value of the weapon system is the incremental difference in the outcomes, V_2-V_1." However, the quantity, V_2-V_1, could well be an overestimation of the effects of the weapon system. The reason is simple. In a conventional tactical campaign, there generally exist alternative strategies for Red that could at least partially counter the new weapon system. Conversely, V_2-V_1 could well be an underestimation or overestimation since there probably exist alternative Blue strategies that would make better use of the new weapon system. Thus, to say that in a conventional tactical campaign (where both forces have a wide range of strategy options) that the value of a weapon system is V_2-V_1, may be an incorrect and misleading statement.

Of course, the problem is the same when alternative Blue forces are compared. The best Red strategy against Blue alternative force A is probably different from the best strategy against Blue alternative B. Similarly, Blue's best strategies will be different.

It becomes obvious then that the relative effectiveness of opposing general-purpose forces should be determined in a context where each is allowed to follow an "optimal" allocation policy which maximizes the force's effectiveness.

Example

This section contains an example which illustrates how a multi-stage simultaneous game can help the decision maker answer the question of which tactical airplane should be purchased with a fixed budget. The example will use the linear model. The country with the procurement decision will play Blue. His enemy is the Red force.

Suppose Red has 100 aircraft with the following capabilities: RABA = 3, RAD = 3, RCAS = 1. Blue may purchase either of two aircraft called type A and type B. Type A has the following capabilities: BABA = 1, BAD = 1, BCAS = 2.

AD-A115 536   AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOO--ETC F/G 15/7
A COMPUTERIZED ALGORITHM FOR SOLVING MULTI-STAGE SIMULTANEOUS G--ETC(U)
DEC 81 M A FATEEN
UNCLASSIFIED AFIT/GOR/MA/81D-5 NL

2 of 2
AD
A115536

END
DATE
FILMED
7-82
DTIC

Type B has capabilities represented by BABA = 1, BAD = 1, BCAS = 3. Thus type B is the better aircraft due to its improved close air support capability. However, the life cycle cost of aircraft type B is 1.25 times the life cycle cost of aircraft type A. Hence, with a fixed budget, more type A aircraft can be purchased.

To analyze this problem, two runs of the linear algorithm were performed--one with the Red aircraft versus Blue type A, and the other with the Red aircraft versus Blue type B. Tables X-1 and X-2 show the results of the first and the second run respectively for up to a 10-stage (day) war. Across the top are the 10 stages and down the left side is the initial force ratio. The tables contain the optimal game value to Blue per Red aircraft.

For example, consider the Red versus Blue type A campaign. Recall that RCAS = 1 and BCAS = 2. The close air support capabilities of the two forces are expressed in terms of equivalent close air support missions. Hence Red aircraft has one equivalent close air support mission and Blue aircraft has two equivalent close air support missions. Suppose that the initial force ratio is 2.0 and the war lasts 5 days. Then, Blue will be able to perform 6.7 more equivalent close air support missions per Red aircraft. Since the initial Red force was 100, at the conclusion of the war, Blue will have performed 670 more equivalent close air support destruction missions against

TABLE X-1

TABLE OF GAME VALUES FOR BLUE TYPE A VERSUS RED

FR	STG1	STG2	STG3	STG4	STG5	STG6	STG7	STG8	STG9	STG10
0.00	-1.00	-2.00	-3.00	-4.00	-5.00	-6.00	-7.00	-8.00	-9.00	-10.00
.05	-.90	-1.80	-2.70	-3.55	-4.47	-5.42	-6.37	-7.33	-8.32	-9.31
.10	-.80	-1.60	-2.47	-3.37	-4.30	-5.27	-6.24	-7.23	-8.22	-9.21
.15	-.70	-1.40	-2.37	-3.26	-4.20	-5.17	-6.14	-7.13	-8.12	-9.11
.20	-.60	-1.27	-2.27	-3.16	-4.10	-5.07	-6.04	-7.03	-8.02	-9.01
.25	-.50	-1.17	-2.17	-3.06	-4.00	-4.97	-5.94	-6.93	-7.92	-8.91
.30	-.40	-1.07	-2.01	-2.89	-3.80	-4.76	-5.73	-6.70	-7.69	-8.68
.40	-.20	-.87	-1.77	-2.64	-3.56	-4.52	-5.49	-6.47	-7.46	-8.45
.50	0.00	-.56	-1.49	-2.37	-3.30	-4.26	-5.24	-6.22	-7.21	-8.20
.75	.50	.00	-.86	-1.73	-2.65	-3.60	-4.53	-5.41	-6.29	-7.17
1.00	1.00	.67	.23	-.19	-.64	-1.11	-1.59	-2.08	-2.59	-3.11
1.25	1.50	1.44	1.15	.81	.45	.11	-.21	-.52	-.81	-1.10
1.50	2.00	2.50	2.95	3.40	3.83	4.29	4.77	5.25	5.74	6.23
1.75	2.50	3.25	3.93	4.59	5.28	6.01	6.76	7.51	8.28	9.05
2.00	3.00	4.00	4.89	5.76	6.70	7.67	8.66	9.64	10.64	11.63
2.50	4.00	6.33	7.91	9.67	11.48	13.36	15.27	17.20	19.16	21.15
3.00	5.00	8.33	11.49	14.59	17.66	20.72	23.78	26.82	29.86	32.89
3.50	6.00	10.06	13.93	17.72	21.49	25.23	28.96	32.69	36.40	40.12
4.00	7.00	11.78	16.34	20.83	25.28	29.71	34.12	38.53	42.92	47.31
4.50	8.00	13.50	18.77	23.96	29.10	34.21	39.31	44.39	49.46	54.53
5.00	9.00	15.22	21.19	27.06	32.88	38.67	44.45	50.21	55.96	61.70
5.50	10.00	16.94	23.62	30.19	36.70	43.18	49.64	56.08	62.50	68.92
6.00	11.00	18.67	26.05	33.33	40.54	47.71	54.86	61.99	69.10	76.20
6.50	12.00	20.39	28.47	36.44	44.34	52.19	60.02	67.83	75.62	83.39
7.00	13.00	22.11	30.90	39.55	48.13	56.67	65.18	73.66	82.13	90.58
7.50	14.00	23.83	33.32	42.66	51.93	61.15	70.33	79.49	88.64	97.76
8.00	15.00	25.56	35.74	45.77	55.72	65.62	75.49	85.33	95.14	104.95
8.50	16.00	27.28	38.16	48.88	59.52	70.10	80.64	91.16	101.66	112.13
9.00	17.00	29.00	40.58	51.99	63.32	74.58	85.80	97.00	108.17	119.32

TABLE X-2

TABLE OF GAME VALUES FOR BLUE TYPE B VERSUS RED

FR	STG1	STG2	STG3	STG4	STG5	STG6	STG7	STG8	STG9	STG10T
0.00	-1.00	-2.00	-3.00	-4.00	-5.00	-6.00	-7.00	-8.00	-9.00	-10.00
.05	-.85	-1.70	-2.55	-3.52	-4.47	-5.42	-6.37	-7.33	-8.31	-9.29
.10	-.70	-1.40	-2.37	-3.37	-4.28	-5.23	-6.20	-7.18	-8.16	-9.14
.15	-.55	-1.22	-2.22	-3.22	-4.13	-5.08	-6.05	-7.03	-8.01	-8.99
.20	-.40	-1.07	-2.07	-3.07	-3.98	-4.93	-5.90	-6.88	-7.86	-8.84
.25	-.25	-.92	-1.92	-2.92	-3.83	-4.78	-5.75	-6.73	-7.71	-8.69
.30	-.10	-.77	-1.75	-2.75	-3.67	-4.60	-5.55	-6.52	-7.49	-8.47
.40	.20	-.47	-1.39	-2.38	-3.30	-4.23	-5.19	-6.16	-7.14	-8.13
.50	.50	-.06	-1.00	-2.00	-2.92	-3.86	-4.82	-5.80	-6.78	-7.76
.75	1.25	.75	-.15	-1.12	-2.04	-2.98	-3.94	-4.91	-5.89	-6.87
1.00	2.00	1.67	1.22	.82	.41	-.02	-.46	-.92	-1.39	-1.87
1.25	2.75	2.69	2.40	2.11	1.78	1.43	1.10	.77	.49	.22
1.50	3.50	4.50	5.43	6.37	7.30	8.21	9.15	10.11	11.07	12.04
1.75	4.25	5.58	6.83	8.06	9.29	10.56	11.86	13.16	14.48	15.81
2.00	5.00	6.67	8.20	9.73	11.29	12.90	14.52	16.16	17.81	19.46
2.50	6.50	10.00	12.32	15.13	17.99	20.92	23.91	26.92	29.95	32.99
3.00	8.00	13.33	18.48	23.58	28.65	33.71	38.76	43.80	48.84	53.87
3.50	9.50	15.89	22.09	28.21	34.30	40.37	46.43	52.48	58.52	64.57
4.00	11.00	18.44	25.67	32.82	39.94	47.03	54.11	61.18	68.24	75.29
4.50	12.50	21.00	29.27	37.45	45.58	53.69	61.78	69.86	77.93	86.00
5.00	14.00	23.56	32.85	42.05	51.21	60.33	69.44	78.53	87.61	96.68
5.50	15.50	26.11	36.45	46.68	56.85	67.00	77.12	87.22	97.31	107.39
6.00	17.00	28.67	40.06	51.34	62.55	73.73	84.88	96.01	107.12	118.22
6.50	18.50	31.22	43.65	55.95	68.18	80.37	92.53	104.68	116.80	128.91
7.00	20.00	33.78	47.23	60.56	73.81	87.02	100.19	113.34	126.48	139.60
7.50	21.50	36.33	50.82	65.17	79.44	93.66	107.85	122.01	136.16	150.29
8.00	23.00	38.89	54.41	69.78	85.07	100.31	115.51	130.68	145.83	160.97
8.50	24.50	41.44	58.00	74.39	90.70	106.95	123.16	139.35	155.51	171.66
9.00	26.00	44.00	61.59	79.00	96.33	113.60	130.82	148.02	165.19	182.35

Red than Red will have performed against Blue. Figures X-1 and X-2 show the same results graphically. The horizontal axis represents the force ratio, the vertical axis the game value. Each curve represents a different number of stages in the war.

Now let's examine the preferences for type A versus type B aircraft for different levels of the budget constraint. Recall that type B aircraft cost 1.25 times as much as type A. Hence, if the budget constraint permits Blue to purchase 50 type A aircraft, the alternative type B purchase would be 40 aircraft. The initial force ratio can be calculated using an initial Red force of 100. The game values in equivalent close air support missions per Red aircraft for a 10-day (stage) war can be found from Tables X-1 and X-2. The results are summarized in Table X-3.

TABLE X-3

PREFERENCE AS A FUNCTION OF BUDGET SIZE

Number of Aircraft Purchased		Initial Force Ratio		Game Value (per Initial Red Aircraft)		
Type A	Type B	Type A	Type B	Type A	Type B	Preference
25	20	.25	.2	-8.91	-8.84	B
50	40	.5	.4	-8.20	-8.13	B
125	100	1.25	1.00	-1.10	-1.87	A
250	200	2.50	2.00	21.15	19.46	A
500	400	.50	4.00	61.70	96.68	B

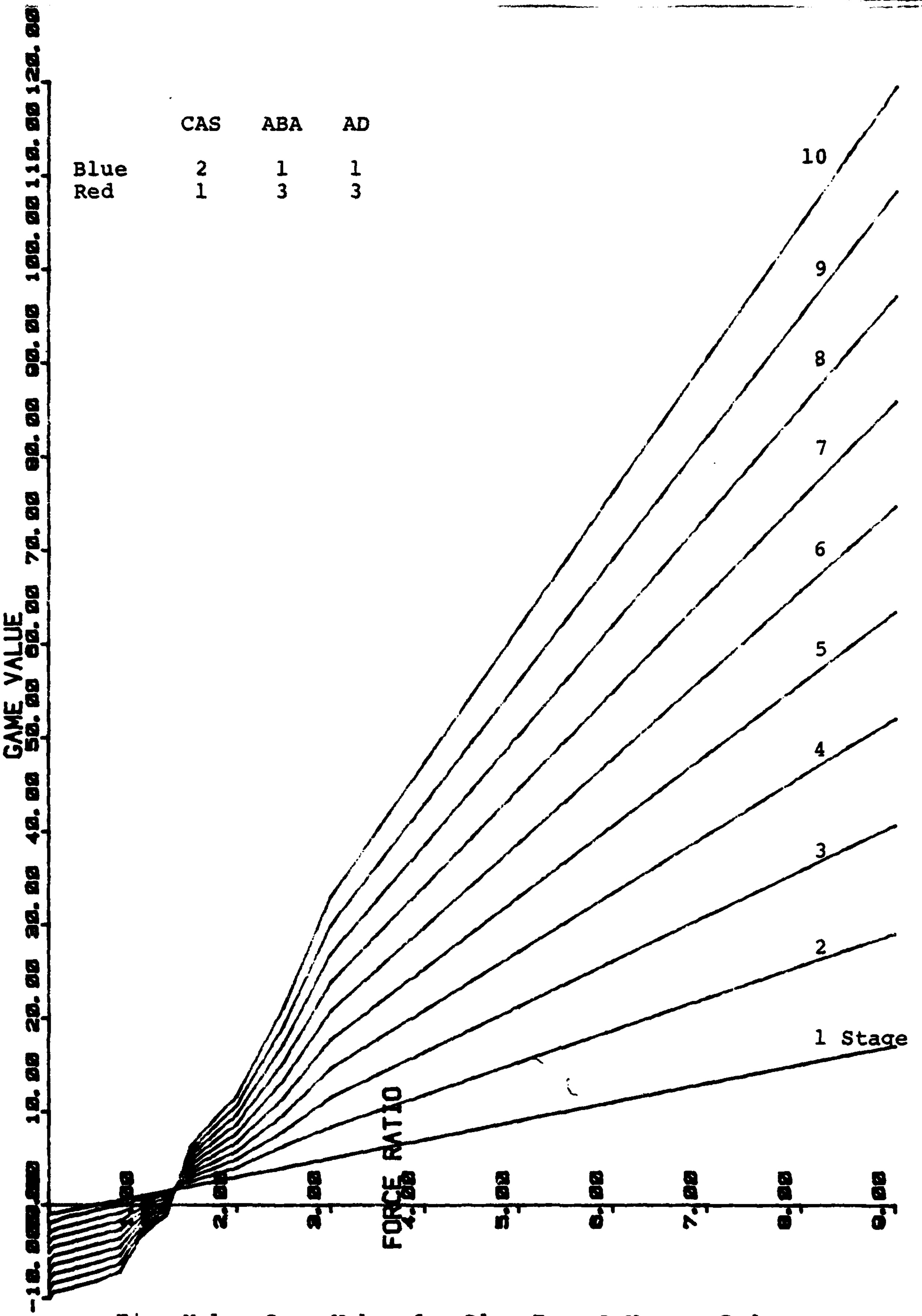


Fig. X-1. Game Value for Blue Type A Versus Red

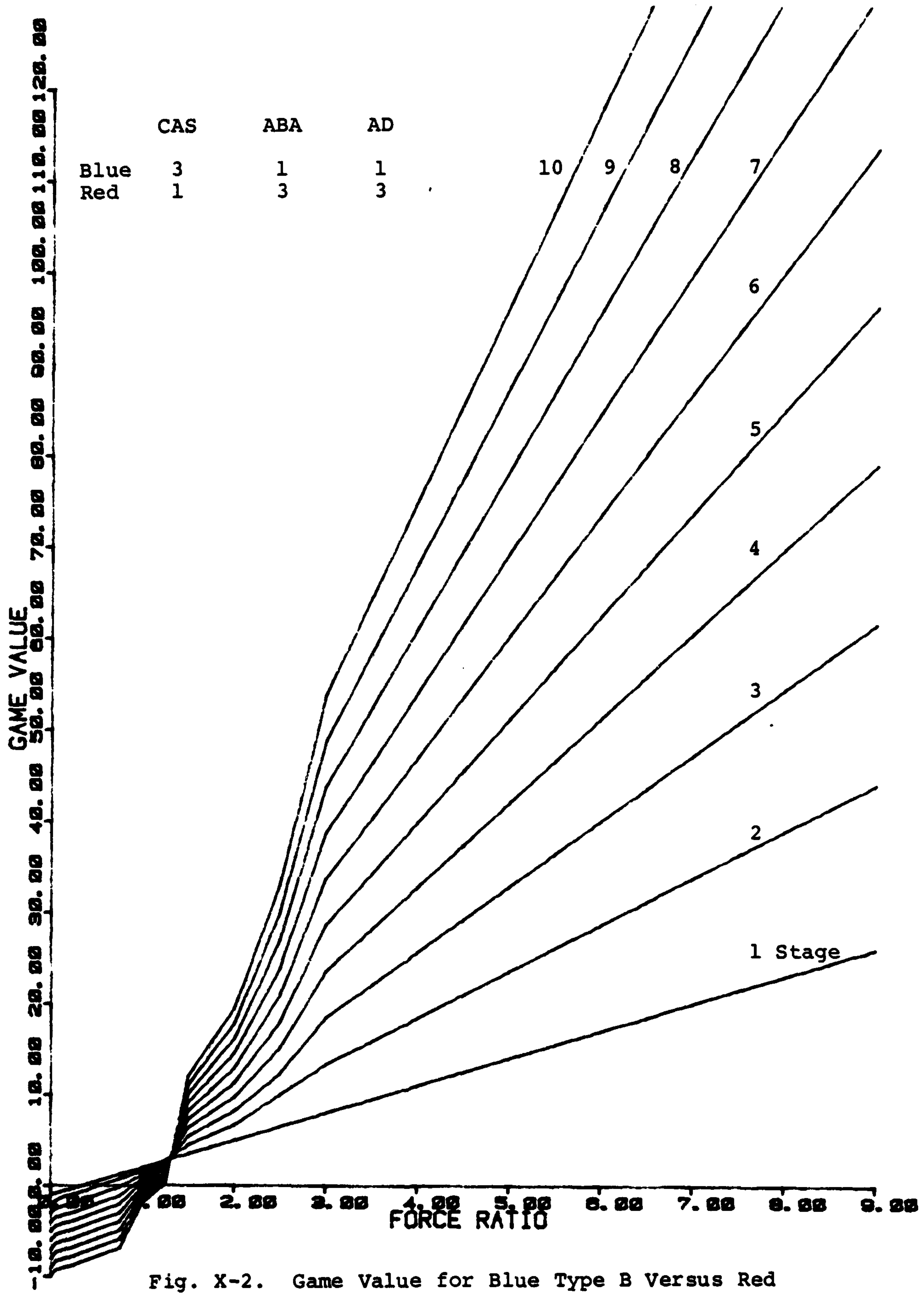


Fig. X-2. Game Value for Blue Type B Versus Red

It is noted that the preferences change based on the initial force ratio. Figure X-3 can be used to determine the force ratios where preference changes occur. The vertical axis continues to be the game value while the upper horizontal axis represents the force ratio for the plot of the Blue type A versus Red and the lower horizontal axis represents the force ratio for the plot of Blue type B versus Red. The dotted curve represents the game value as a function of force ratio for the Blue type B versus Red 10-day campaign. Initially, type B is preferred to Type A. When the type A to Red force ratio reaches about .75 then type A is preferred. Another cross-over occurs when the type A to Red force ratio is 1.7 and 2.2. A final cross-over occurs when the type A to Red force ratio exceeds 3.3. For all type A to Red force ratios in excess of 3.3 Blue type B is the preferred purchase. Table X-4 shows that the preference can also vary depending on the number of stages in the war. Assume the budget allows the purchase of 125 type A or 100 type B Blue aircraft.

Finally, Figure X-4 illustrates this concept for the ten stages.

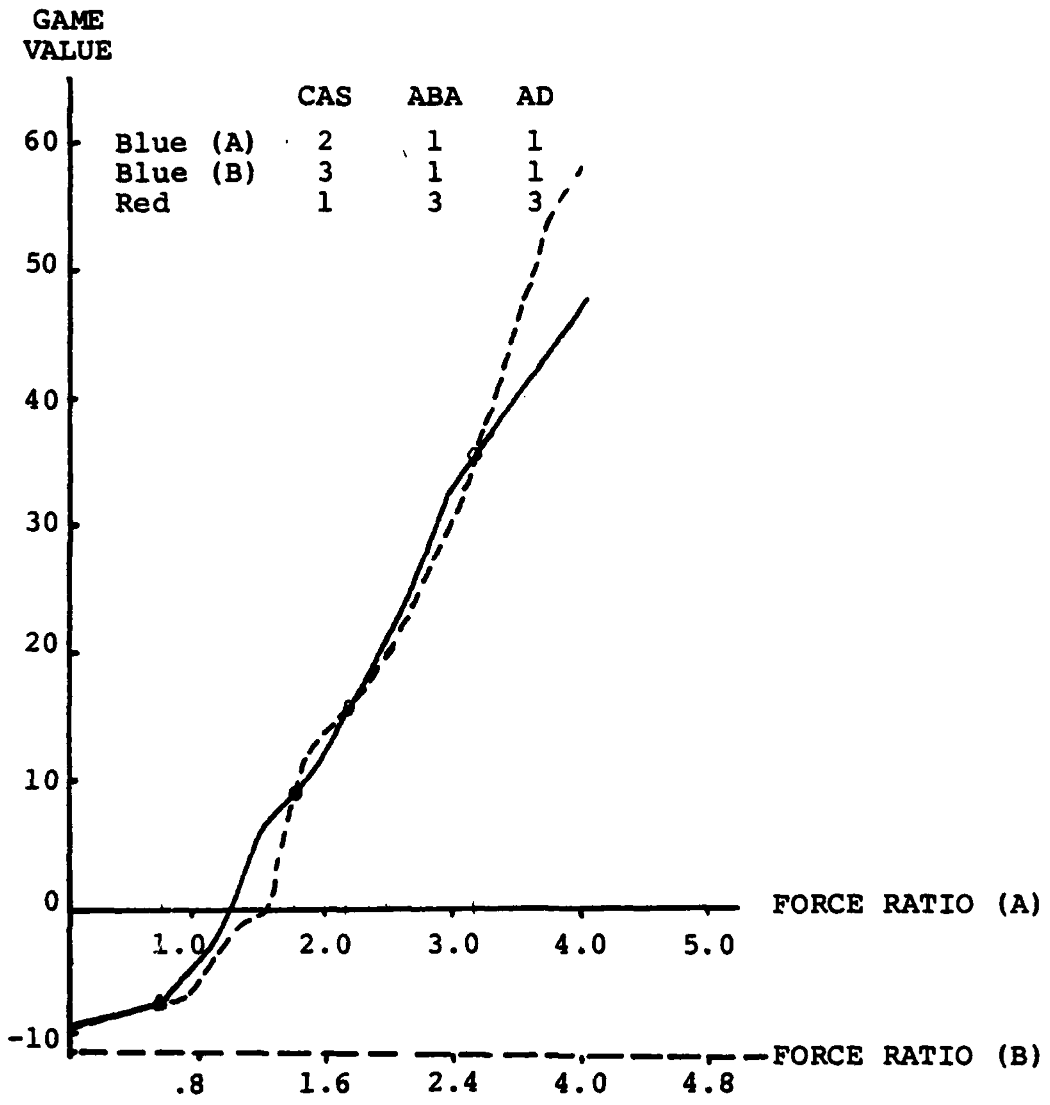


Fig. X-3. The Tenth Stage Game Value of Blue Type A (Solid) and Blue Type B (Dotted) Versus Red

TABLE X-4

PREFERENCE AS A FUNCTION OF NUMBER OF STAGES IN WAR

Number of Stages	Game Value Type A	Game Value Type B	Preference
1	1.5	2.0	B
2	1.44	1.67	B
3	1.15	1.22	B
4	.81	.82	B
5	.45	.41	A
6	.11	-0.02	A
7	-.21	-.46	A
8	-.52	-.92	A
9	-.81	-1.39	A
10	-1.10	-1.87	A

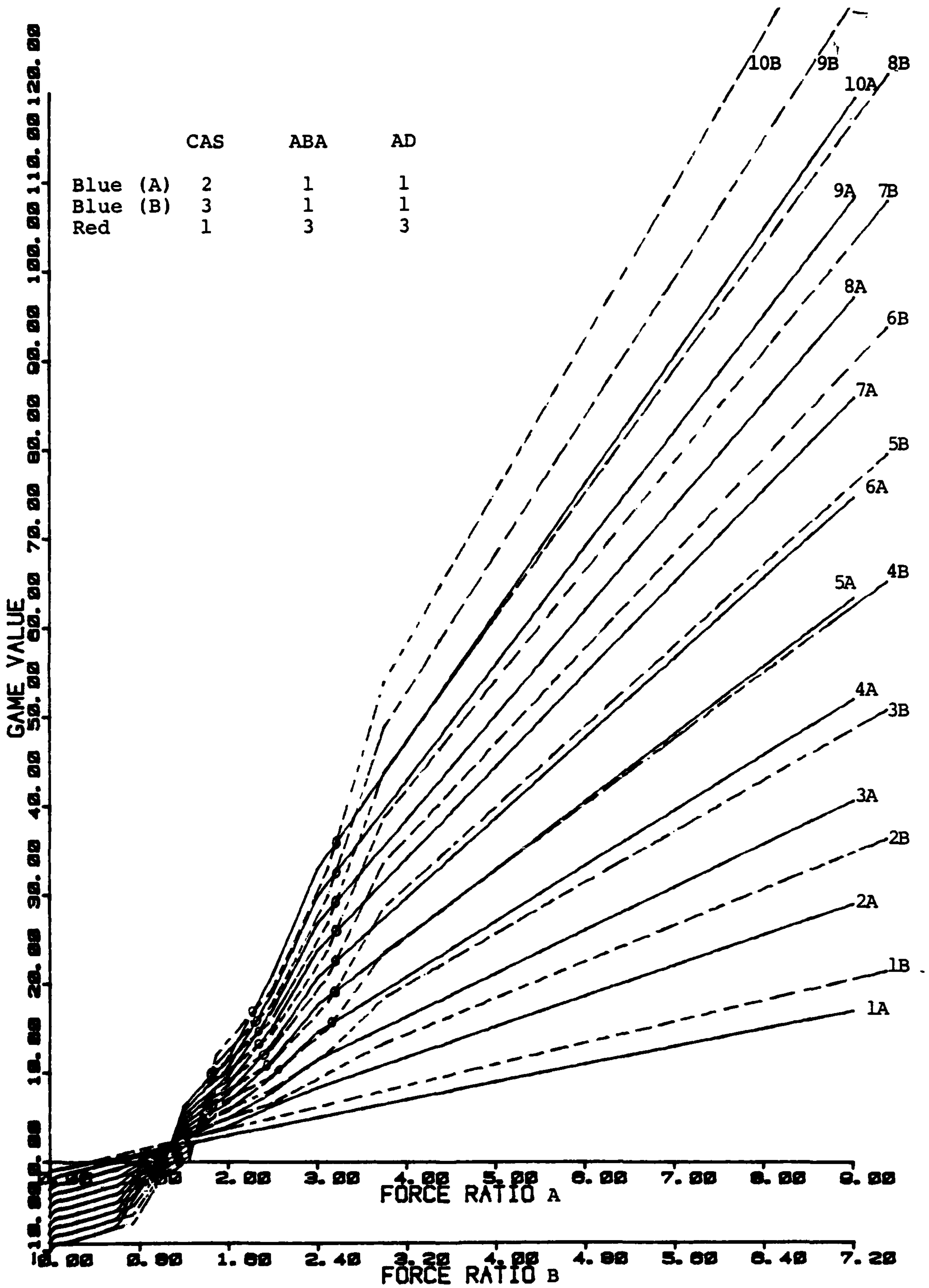


Fig. X-4. Game Value for Blue Type A (Solid) and Blue Type B (Dotted) Versus Red

XI. A Modification to the Linear Model

Introduction

The linear algorithm does not allow for air-to-air attrition. In this chapter, a binomial probability distribution is used to represent

1. the probability that an air base defender kills an air base attacker,

2. the probability an air base attacker that survives the initial attack by the air base defender is able to return fire and kill the air base defender, and

3. the probability that a weapon dropped by an air base attacker that is not engaged by an air base defender kills an aircraft caught on the ground. The air base attacker is assumed to carry M weapons (bombs).

Formulation of the Binomial Kill Probabilities

Consider a one-sided combat between two homogeneous forces, a force of N indistinguishable "targets" and a force of Z indistinguishable "interceptors" each having M shots. Suppose the following assumptions apply:

1. At a fixed time the locations of all N targets are known by the interceptors and hence are vulnerable to intercept.

2. Each interceptor can attack only one target in air-to-air combat ($M=1$) but it can attack up to M targets in ground attack.

3. The Z interceptors decide to allocate their shots among the N targets as uniformly as possible to maximize the marginal expected value destroyed. If $Z \times M$ is an integral multiple of N, then each target will be assigned exactly $L=(Z \times M/M)$ shots, and the probability of survival of each target will be equal to SUR(L) where $SUR(L) = (1-p)^L$.

If $Z \times M$ is not an integral multiple of N, then we can express the ratio $L = (Z \times M/N)$ as

$$L = I + F \qquad (11\text{-}1)$$

where I denotes the largest integer smaller than the ratio L, and F denotes the fractional part of the ratio. In this case the optimal targeting will be to allocate (I+1) shots to a fraction F of the targets and I shots to the rest. Thus, the average survival probability is given by SUR(L), where

$$SUR(L) = (1-F)(1-p)^{I} + F(1-p)^{I+1} = (1-p)^{I}\ (1-pF) \qquad (11\text{-}2)$$

For L less than unity $z \times M < N$, then $I=0$, and SUR(L) can be reduced to the simple form

$$SUR(L) = 1 - (p \times Z \times M/N) \qquad \text{for } Z \times M < N \qquad (11\text{-}3)$$

Let K denote the number of targets killed and E(K) denote the expected number of targets killed, then

$$E(K) = N(1-SUR(L))$$

$$= \begin{cases} p \times Z \times M & \text{for } Z \times N \leq N \\ N[1-(1-p)^{I}(1-pF)] & \text{for } Z \times N \leq N \end{cases} \qquad (11\text{-}4)$$

The binomial equation (11-4) will be used in the binomial model of air war.

<u>The Binomial Model--Definition of the Input Quantities</u>

$BLUF^{(ISTAG)}$	= total number of Blue aircraft at the beginning of stage ISTAG.
$REDF^{(ISTAG)}$	= total number of Red aircraft at the beginning of stage ISTAG.
$ABAB^{(ISTAG)}$	= number of Blue aircraft sent on air base attack at stage ISTAG.
$ABAR^{(ISTAG)}$	= number of Red aircraft sent on air base attack at stage ISTAG.
$ABD^{(ISTAG)}$	= number of Blue aircraft sent on air defense at stage ISTAG.
$ADR^{(ISTAG)}$	= number of Red aircraft sent on air defense at stage ISTAG.
$CASB^{(ISTAG)}$	= number of Blue aircraft sent on close air support at stage ISTAG.

$CASR^{(ISTAG)}$	= number of Red aircraft sent on close air support at stage ISTAG.
RDTBA	= the ratio between Red defenders and Blue attackers.
BDTRA	= the ratio between Blue defenders and Red attackers.
IRDTBA	= the largest integer less than or equal to RDTBA.
IBDTRA	= the largest integer less than or equal to BDTRA.
RDKBA	= probability that a given Red defender will kill the Blue attacker to which he was assigned.
BAK	= expected number of Blue attackers killed.
BAJ	= number of Blue attackers that jettison their load.
BAKRD	= probability that a given Blue attacker that survives the Red defender's first round and jettisons his load will kill a Red defender.
RDK	= expected number of the Red defenders killed.
BDKRA	= probability that a given Blue defender kills one Red attacker.
RAK	= expected number of Red attackers killed.

RAJ	= number of Red attackers that jettison their load.
RAKBD	= probability that a Red attacker that survives the Blue defender's first round and jettisons his load will kill a Blue defender.
BDK	= expected number of Blue defenders killed.
BLGR	= expected number of surviving Blue aircraft after the air-to-air engagement.
RDGR	= expected number of surviving Red aircraft after the air-to-air engagement.
BLABA	= number of Blue attackers which remain in their mission and penetrate after air-to-air engagement.
RDABA	= number of Red attackers which remain in their mission and penetrate after air-to-air engagement.
FBC	= fraction of the Blue aircraft caught on the ground.
FRC	= fraction of the Red aircraft caught on the ground.
BABA	= number of Blue shots which can be used to kill Red aircraft on the ground.
BABAP	= probability that a given shot by a Blue penetrator will kill one Red aircraft on the ground.

RABA	= number of Red shots which can be used to kill Blue aircraft on the ground.
RABAP	= probability that a given shot by a Red penetrator will kill one Blue aircraft on the ground.
BASTRF	= the ratio between the number of shots Blue penetrators have and the number of Red aircraft caught on the ground.
RASTBF	= the ratio between the number of shots Red penetrators have and the number of Blue aircraft caught on the ground.
IBASTRF	= the largest integer less than or equal to BASTRF.
IRSTBF	= the largest integer less than or equal to RASTBF.
BGK	= expected number of Blue aircraft killed on the ground.
RGK	= expected number of Red aircraft killed on the ground.
BCAS	= the close air support capability of a Blue aircraft.
RCAS	= the close air support capability of a Red aircraft.

Computation of the Binomial Model

Air to Air Destruction. The following equations are used to calculate the results of the air-to-air engagement.

Red defender to Blue attacker ratio

$$RDTBA = \frac{ADR^{(ISTAG)}}{ABAB^{(ISTAG)}} \qquad (11\text{-}5)$$

Integer of Red defender to Blue attacker ratio

$$IRDTBA = [RDTBA] \qquad (11\text{-}6)$$

Blue attacker killed

$$BAK = ABAB^{(ISTAG)}[\{(1-RDKBA)^{IRDTBA}(1-RDKBA[RDTBA-IRDTBA])\}] \qquad (11\text{-}7)$$

Blue attacker jettison

$$BAJ = \min[ABAB^{(ISTAG)}, ADR^{(ISTAG)}] \qquad (11\text{-}8)$$

Red defender killed

$$RDK = (BAJ-BAK)BAKRD \qquad (11\text{-}9)$$

Blue defender to Red attacker ratio

$$BDTRA = \frac{ADB^{(ISTAG)}}{ABAR^{(ISTAG)}} \qquad (11\text{-}10)$$

Integer of Blue defender to Blue attacker ratio

$$IBDTRA = [BDTRA] \qquad (11\text{-}11)$$

Red attacker killed

$$\mathrm{RAK} = \mathrm{ABAR}^{(\mathrm{ISTAG})}\ [1-\{(1-\mathrm{BDKRA})^{\mathrm{IBATRA}}\ (1-\mathrm{BDKRA}[\mathrm{BDTRA}-\mathrm{IBDTRA}[\,)\}] \qquad (11\text{-}12)$$

Red attacker jettison

$$\mathrm{RAJ} = \min[\mathrm{ABAR}^{(\mathrm{ISTAG})},\ \mathrm{ADB}^{(\mathrm{ISTAG})}] \qquad (11\text{-}13)$$

Blue defender killed

$$\mathrm{BDK} = (\mathrm{RAJ}-\mathrm{RAK})\mathrm{BAKRD} \qquad (11\text{-}14)$$

Blue force after air-to-air engagements

$$\mathrm{BLGR} = \mathrm{BLUF}^{(\mathrm{ISTAG})} - \mathrm{BAK} - \mathrm{BDK} \qquad (11\text{-}15)$$

Red force after air-to-air engagements

$$\mathrm{RDGR} = \mathrm{REDF}^{(\mathrm{ISTAG})} - \mathrm{RAK} - \mathrm{RDK} \qquad (11\text{-}16)$$

Air Base Destruction. The following equations are used to calculate the number of aircraft destroyed on the ground for each side.

Blue attackers penetrated

$$\mathrm{BLABA} = \mathrm{ABAB}^{(\mathrm{ISTAG})} - \mathrm{BAJ} \qquad (11\text{-}17)$$

Red attackers penetrated

$$\mathrm{RDABA} - \mathrm{ABAR}^{(\mathrm{ISTAG})} - \mathrm{RAJ} \qquad (11\text{-}18)$$

Ratio between Blue penetrator shots and the Red force caught on the ground

$$BASTRF = \frac{BLABA \times BABA}{FBL \times RDGR} \qquad (11\text{-}19)$$

Integer of BASTRF

$$IBASTRF = [BASTRF] \qquad (11\text{-}20)$$

Blue aircraft killed on the ground

$$BGK = FBC \times BLGR\ [1-\{(1-RABAP)^{IRSTBF}\ (1-RABAP[RSTBF-IRSTBF])\}] \qquad (11\text{-}21)$$

Red aircraft killed on the ground

$$RGK = FRC \times RDGR\ [1-\{(1-BABAP)^{IBSTRF}\ (1-BABAP[BSTRF-IBSTRF])\}] \qquad (11\text{-}22)$$

Close Air Support.

$$PAYOFF^{(ISTAG)} = CASB^{(ISTAG)} \times BCAS - CASR^{(ISTAG)} \times RCAS \qquad (11\text{-}23)$$

The Remaining Forces to (ISTAG-1). Finally, the total number of surviving forces for each side is calculated.

$$BLUF^{(ISTAG-1)} = BLUF^{(ISTAG)} - BAK - BDK - BGK \qquad (11\text{-}24)$$

$$REDF^{(ISTAG-1)} = REDF^{(ISTAG)} - RAK - RDK - RGK \qquad (11\text{-}25)$$

One Stage Equations for the Binomial Model

The state variables $BLUF^{(ISTAG)}$ and $REDF^{(ISTAG)}$ can again be transformed to the force ratio $BI^{(ISTAG)}$ and the initial red force size $REDF^{(MSTAG)}$. As in Chapter VII, we define s(J,1), s(J,2), and s(J,3) as the fraction of Blue force assigned to close air support, air base attack and air defense, respectively. Similarly, s(K,1), s(K,2), and s(K,3) are the fraction of Red force assigned to close air support, air base attack and air defense, respectively. We can use equations 11-5 to 11-24 with the force ratio and initial Red force state variables. Replace the "number" in equations 11-5 to 11-24 with the force ratio and $REDF^{(ISTAG)}$. $BLUF^{(ISTAG)}$ will then be the force ratio $BI^{(ISTAG)}$ when $REDF^{(ISTAG)}$ equals one. In addition,

$$ABAB^{(ISTAG)} = BI^{(ISTAG)} \times s(J,2)$$

$$ABAR^{(ISTAG)} = s(K,2)$$

$$ADB^{(ISTAG)} = BI^{(ISTAG)} \times s(J,3)$$

$$ADR^{(ISTAG)} = s(K,3)$$

$$CASB^{(ISTAG)} = BI^{(ISTAG)} \times s(J,1)$$

$$CASR^{(ISTAG)} = s(K,1)$$

Equation 11-24 then becomes

$$BI^{(ISTAG-1)} = BI^{(ISTAG)} - BAK - BDK - BGK \qquad (11\text{-}26)$$

Equation 11-24 becomes

$$RR^{(ISTAG)} = 1 - RAK - RDR - RGX \qquad (11\text{-}27)$$

The payoff for this stage when Blue plays his Jth strategy and Red plays his Kth strategy, assuming that $REDF^{(ISTAG)}$ is one, is

$$T(BI^{(ISTAG)},J,K) = BI^{(ISTAG)} \times s(J,1) \times BCAS$$
$$- s(K,1) \times RCAS \qquad (11\text{-}28)$$

By using this notation, equation (7-13) can be used to express the total payoff Blue wants to maximize.

$$M = REDF^{(MSTAGE)} [T(BI^{(MSTAGE)},J,K) + RR^{(MSTAGE)}$$
$$[T(BI^{(MSTAG-1)},J,K + RR^{(MSTAGE-1)} [\ldots$$
$$+ RR^{(2)} [T(BI^{(1)},J,K]]\ldots] \qquad (11\text{-}29)$$

The dynamic programming approach developed in Chapter VII can now be used to obtain a numerical solution of this problem. Time did not permit the computer implementation of this approach.

XII. Summary, Conclusions, and Recommendation

The objectives of this study, as listed in Chapter I, have been accomplished. An algorithm was developed to solve the multi-stage simultaneous game.

A multi-stage simultaneous game consists of many two-person zero-sum games (stages). The payoffs on each stage are determined by the strategies chosen by the players and by a state vector describing the capabilities of each side at the beginning of the stage. A state transition function is used to determine the state vector of capabilities at the next stage from the state vector for the previous stage and the player's strategies. The main idea of the solution algorithm is to build at each stage a matrix whose values are the payoffs obtained by playing each of the given strategies at this stage and the optimal strategies for the remaining stages. A dynamic programming approach is used to determine the payoff obtained by using the optimal strategies for the remaining stages. Expected game values are calculated at each user-specified grid point. A cubic spline interpolation is used to find expected game values at intermediate points.

A tactical air war problem was selected to illustrate the usefulness of the algorithm. The missions are close air support, air base attack and air defense. A linear model was developed in which the state vector (number of Red and Blue aircraft) could change from stage to stage only when air base attackers successfully penetrate the other side's air defense and kill aircraft on the ground. Air defense can prevent the other side from attacking the air base but will not result in any force attrition. Rather than using the force sizes of Blue and Red as the state variables in the computerized algorithm, the author proved that a constant (initial Red force size) and the force ratio (Blue/Red) could be used to completely describe the state of the game. This reduces the size of the problem by the square root (e.g., a problem using 100 grid points with two state variables can be solved with the same accuracy using 10 grid points and a single state variable).

A "GAME" was developed for training and assisting the field commander. In the assist mode, the GAME is able to answer the "what if" questions of the commander. In the training mode, the GAME begins in any given initial condition, the computer plays optimally as one side and the commander plays the other side. At each stage the state variable is updated according to the strategies chosen by the player and the computer. The player has the option of asking the GAME to tell him his optimal mixed strategies.

The GAME uses an output of the linear model, which gives the optimal game value at any stage and at any state.

Then GAME can calculate the optimal strategies for both sides for any given conditions by solving only one payoff matrix.

To compare two weapon systems, and to be fair in the comparison, one should let each system play its optimal strategies. The multi-stage simultaneous game is an excellent way to compare two weapon systems when each side plays its optimal strategy. An example comparing two types of aircraft is shown in Chapter X. In this example, the preferred system depends on the number of days (stages) in the campaign and the initial force ratio.

A binomial model that permits attrition to both sides in the air-to-air engagements is developed in Chapter XI. This model can also use the force ratio concept.

For an extension of this effort the author recommends the following:

1. Preparing a computer code for the binomial model and comparing this model to the linear model.

2. Using more missions, such as reconnaissance and SAM suppression.

3. Using more than one type of aircraft.

4. Examining other potential applications of the algorithm to various types of games in war and in business.

5. Using the GAME as a method of introducing multi-stage game theory to students in operations research.

The GAME can help the students understand the concepts of game value and mixed strategies.

6. Comparing actual weapon systems using the multi-stage game algorithm. Determine how much help they may be in the procurement decision.

In conclusion, this research has provided the author and, hence, the Egyptian Air Force with a valuable tool which hopefully will be used to both help train Egyptian Air Force commanders and help make important procurement decisions.

Appendix A

Interpolation With Cubic Spline Functions

One of the difficulties with conventional polynomial interpolation, particularly if the polynomial is of high order, is the highly inflected or "wiggly" character which it is possible for the interpolating polynomial to assume.

A smoother interpolating function can be produced using the cubic spline function.

The construction of a cubic spline interpolating function can be briefly described as follows (8 :47). We are given a series of points $x_i (i=0,1,2,...,n)$ which are in general not evenly spaced and the corresponding function values $F(x_i)$. Now consider two arbitrary adjacent points x_i and x_{i+1}. We wish to fit a cubic to these two points and use this cubic as the interpolating function between them. We denote this cubic as

$$F_i(x) = a_0+a_1x+a_2x^2+a_3x^3 \quad (x_1 \leq x \leq x_{i+1}) \quad \text{(A-1)}$$

There are four unknown constraints in (i), and only two conditions are immediately obvious, namely that $F_i(x_i) = f(x_i)$ and $F_i(x_{i+1}) = f(x_{i+1})$. We are free to choose the remaining conditions as we like, to accomplish our desired objective of "smoothness." The most effective approach is to match the first and second derivatives (and thus the slope and curvature) of $F_i(x)$ to those of the cubic $F_{i-1}(x)$ used for interpolation on the adjacent interval $x_{i-1} \leq x \leq x_i$. If this procedure is carried out for all intervals in the region $x_0 \leq x \leq x_n$ (with special treatment at the end points) then an approximating function for the region will have been constructed, consisting of the set of cubic functions, $F_i(x)$ $(i=0,1,...,n-1)$.

Appendix B

Dynamic Programming

Dynamic programming is a mathematical technique often useful for making a sequence of interrelated decisions. It requires formulating an appropriate recursive relationship for each individual problem. However, it provides a large computational saving over using exhaustive enumeration to find the best combination of decisions, especially for large problems.

The basic features which characterize dynamic programming problems are presented below (Ref 7:270-271).

1. The problem can be divided into stages, with a policy decision required at each stage.

2. Each stage has a number of stages associated with it.

3. The effect of the policy decision at each stage is to transform the current state into a state associated with the next stage.

4. Given the current state, an optimal policy for the remaining stages is independent of the policy adopted in previous stages.

5. The solution procedure begins by finding the optimal policy for each state of the last stage.

6. A recursive relationship that identifies the optimal policy for each state at stage n, given the optimal policy for each state at stage (n+1), is available.

7. Using this recursive relationship, the solution procedure moves backward stage by stage--each time finding the optimal policy for each state of that stage--until it finds the optimal policy when starting at the initial stage.

Appendix C

User's Guide for the Game

This section is designed to acquaint the player with the rules, procedures, and peculiarities of the game. The game was designed to be played interactively and was implemented in FORTRAN IV on the CDC CYBER 175 INTERCOM system. The program F3GAME must be edited before the GAME program (see Figure C-1) can be run. The program F3GAME will request the player to set the maximum number of stages to be played, the aircraft capabilities and the number of grid points. F3GAME will calculate the optimal game value for each grid point and for each stage and will write it on TAPE3. When the ROW of F3GAME finishes, the player should EDIT GAME and run it.

GAME will do the following steps:

1. Rewind tape 3 and read the data from it.
2. The program will print

```
"YOU CAN PLAY FROM 1 TO # STAGES"
"HOW MANY STAGES DO YOU WANT TO PLAY?"
```

where # is the maximum number of stages already set on F3GAME. The player has to choose the number of stages he wants to play.

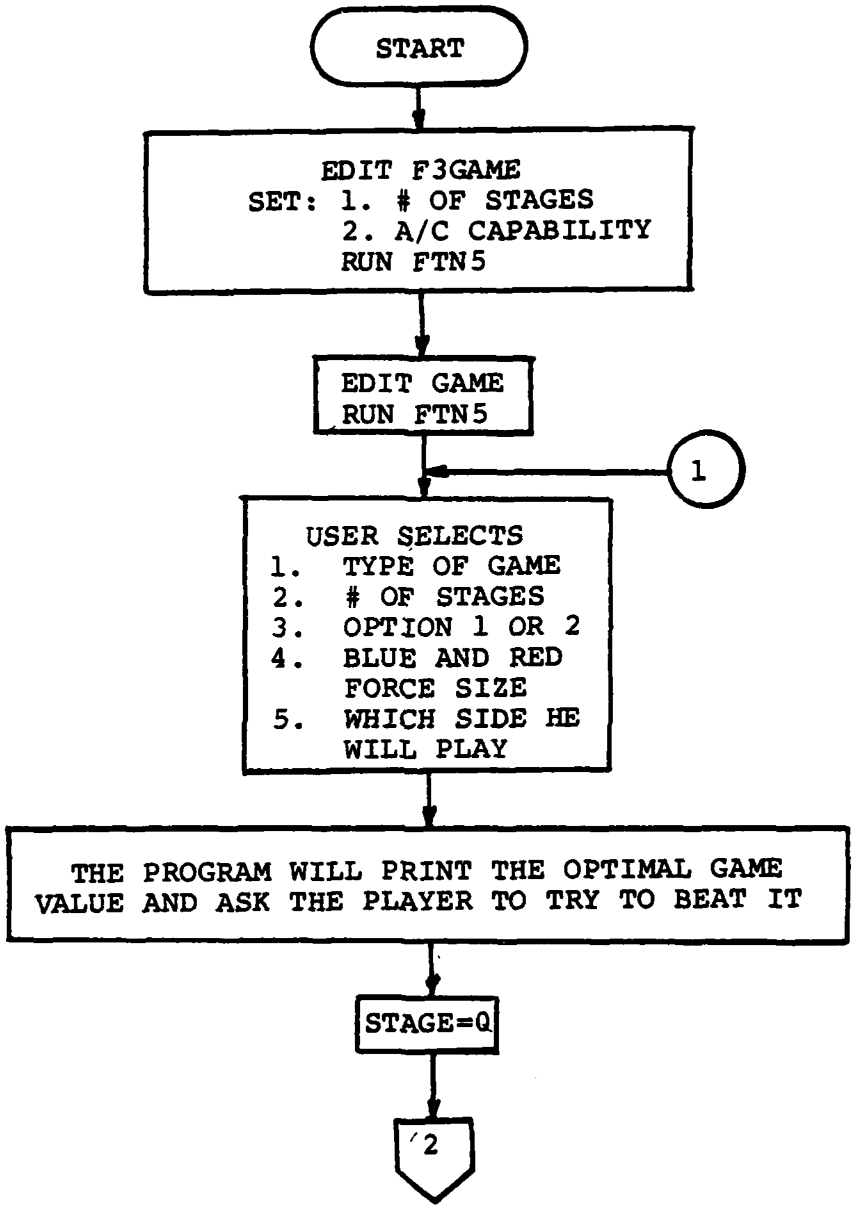


Fig. C-1. Playing Sequence Diagram

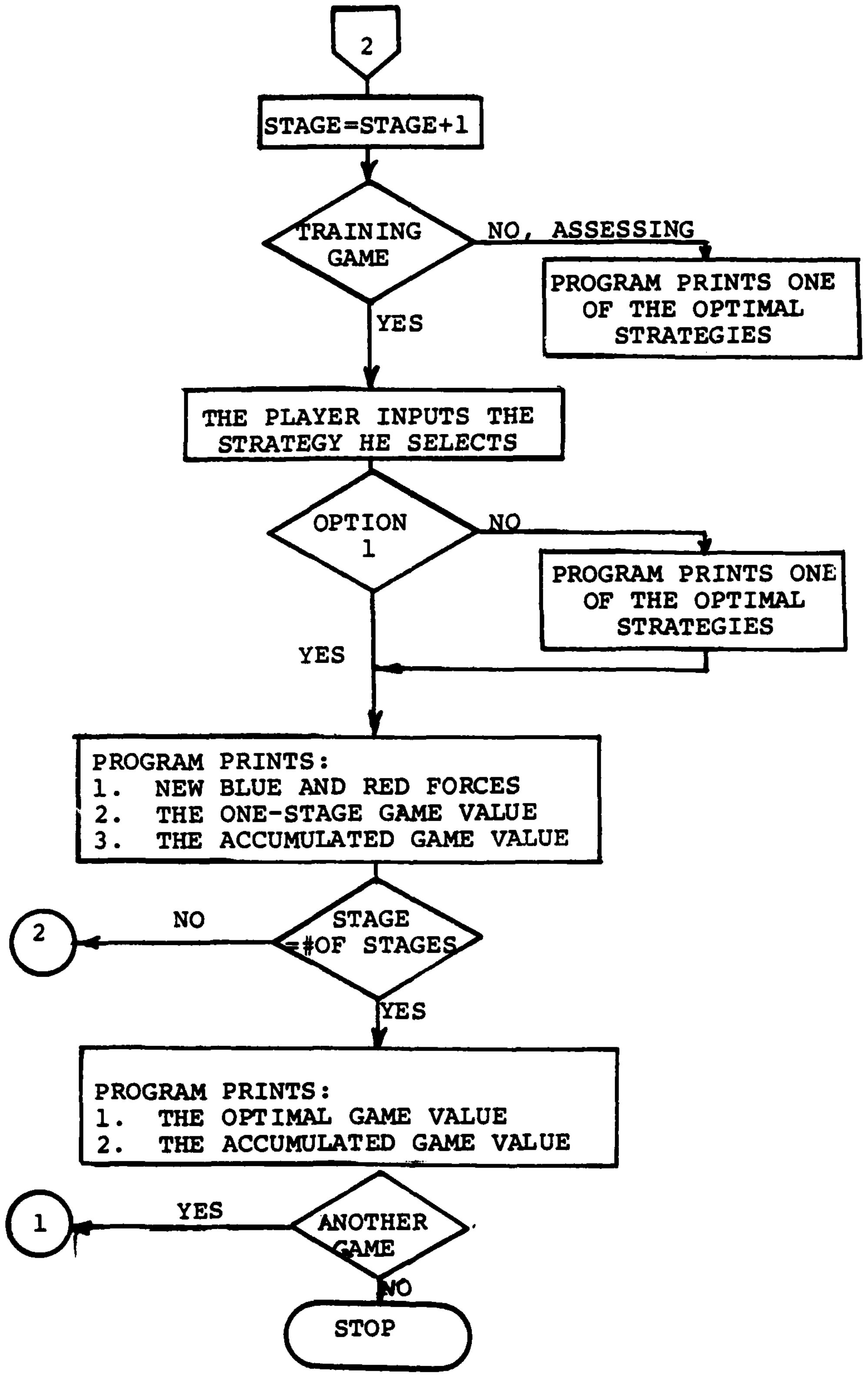

2
STAGE=STAGE+1
TRAINING GAME
NO, ASSESSING
PROGRAM PRINTS ONE OF THE OPTIMAL STRATEGIES
YES
THE PLAYER INPUTS THE STRATEGY HE SELECTS
OPTION 1
NO
PROGRAM PRINTS ONE OF THE OPTIMAL STRATEGIES
YES
PROGRAM PRINTS:
1. NEW BLUE AND RED FORCES
2. THE ONE-STAGE GAME VALUE
3. THE ACCUMULATED GAME VALUE
NO
2
STAGE =#OF STAGES
YES
PROGRAM PRINTS:
1. THE OPTIMAL GAME VALUE
2. THE ACCUMULATED GAME VALUE
YES
1
ANOTHER GAME
NO
STOP

Fig. C-1--*Continued*

3. The program will print

"YOU HAVE THE OPTIONS:"

"1. NOT TO SEE THE OPTIMAL SOLUTION."

"2. TO SEE ONE OF THE OPTIMAL SOLUTIONS."

The player selects option 1 or option 2 by printing "1" or "2".

4. The program will ask the player to "PRINT: BLUE FORCE, RED FORCE" and the player prints the number of the Blue aircraft and the number of the Red aircraft.

5. The program will print

"IF COMPUTER PLAYS BLUE PRINT 1"

"IF COMPUTER PLAYS RED PRINT 11"

and the player has to select 1 or 11.

6. Depending on the force ratio, the number of stages, and the data given by TAPE2 the program will calculate the optimal game value and the optimal mixed strategies for both sides at this stage. The program then will draw a random number and decide which strategy it will play. At the first stage the program will print

"TRY TO BEAT TOTAL GAME VALUE OF #"

where # is the optimal game value calculated before.

7. The program will list the different strategies and the different capabilities for both sides. The result is:

STRATEGY #	1	2	3	4	5	6	7	8	9	10	BLUE	RED
CAS	3	2	2	1	1	1	0	0	0	0	CASB	CASR
ABA	0	1	0	2	1	0	3	2	1	6	ABAB	ABAR
AD	0	0	1	0	1	2	0	1	2	3	ADB	ADR

where CASB, CASR, ABAB, ABAR, ADB, ADR are the capabilities already set on F3GAME. For example:

8. The program will ask the player to print his strategy. Then if option 2 was chosen, the program will show the optimal probability of choosing each of the strategies (mixed strategy).

9. As a result of strategies chosen by both sides, the program calculates and prints new BLUE FORCE totals (numbers), new RED FORCE totals, GAME VALUE, and total GAME VALUE.

10. Steps 6-9 are repeated for each of the stages assigned.

11. At the end of the game the optimal game value is printed with the total game value for comparison.

12. If the player wants to play another game, the program returns to step 2.

The following is the computer code of the GAME program.

```
      PROGRAM GAME
       REAL TT(10,10,2),Z,STR(40),X(40),Y(400)   , B(12),C(10),
     :V,PSOL(10),DSOL(12),RW(178) ,YY(40) ,BI ,A(12,10),T
      INTEGER IA,N,M1,M2,IW(40),IER,NSTAGE,L,M ,I,J,K ,IS

      INTEGER IA,N,M1,M2,IW(40),IER,NSTAGE,L,M ,I,J,K ,ISTAG,SS(1
0,3)
        DOUBLE PRECISION NL
       REWIND 3
      READ (3,*) NSTAGE,RABA,BABA,RAD,BAD,RCAS,BCAS,NF
      DATA SS/3,2,2,1,1,1,4*0,0,1,0,2,1,0,3,2,1,0,0,0,1,0,1,2,0,1
,2,
     :3/,M1/10/,M2/0/,N/10/,IA/12/,B/12*1/,C/10*1/
        NL=123.D0
         IY=0
        DO 20 IS=1,NSTAGE
        READ (3,*) (YY(II),II=1,NF)
        DO 25 II=1,NF
         IY=IY+1
25      Y(IY)=YY(II)
20      CONTINUE
        READ (3,*) (X(II),II=1,NF)
2000    PRINT *, 'YOU CAN PLAY FROM 1 TO ',NSTAGE,' STAGES'
        PRINT *,'  HOW MANY STAGES YOU WANT TO PLAY ?  '
      PRINT *
         READ *,NST
        PRINT *,' YOU HAVE THE OPTIONS :'
        PRINT *, ' 1- NOT TO SEE THE OPTIMAL SOLUTION'
        PRINT *,' 2- TO SEE THE OPTIMAL SOLUTION AFTER YOU PLAY'
         PRINT *,' WHICH OPTION YOU CHOSE?  '
          READ *,NOP
          PRINT *, ' PRINT : BLUE FORCE , RED FORCE '
          PRINT *
          READ *, BLUF,REDF
          BI= BLUF/REDF
         TG=0
        PRINT *,' IF COMPUTER PLAY BLUE PRINT 1'
        PRINT *,' IF COMPUTER PLAY RED PRINT 11'
           NN=0
      PRINT *
        READ *, IPL
       DO 1000 ISTAG=NST,1,-1
       NN=NN+1
       PRINT *, 'STAGE #',NN
        Z=0
```

```
         IF (( BLUF.EQ.0.).OR.(REDF.EQ.0.))THEN
          NPL=IPL
         ELSE
         BI=BLUF/MAX(10.**(-10),REDF)
          DO 800 J=1,10
          DO 800 K=1,10
            TT(J,K,1)=(SS(J,1)*BI*BCAS/3.-SS(K,1)*RCAS/3.)
         IF (ISTAG.EQ.1) THEN
            A(J,K)=TT(J,K,1)
            A(J+2,K)=0
            Z=MIN(Z,A(J,K))
          ELSE
            RED=MAX(0.,SS(J,2)*BI*BABA/3.-SS(K,3)*RAD/3.)
            BLUE=MAX(0.,SS(K,2)*RABA/3.-SS(J,3)*BI*BAD/3.)
            T2=MAX(10.**(-10),(1-RED))
            T=MAX(10.**(-10),(BI-BLUE))/MAX(10.**(-10),(1-RED))
             DO 30 II=1,NF
              IF(X(II).GE.T ) THEN
           IY=II+(ISTAG-2)*NF
               PAY=Y(IY-1)+(Y(IY)-Y(IY-1))*(T-X(II-1))/(X(II)-X(II-1))
               GO TO 100
              ENDIF
 30          CONTINUE
            IY=ISTAG*NF
             PAY=Y(IY)+(Y(IY)-Y(IY-1))*(T-X(NF))/.5
 100         TT(J,K,2)=TT(J,K,1)+PAY*T2
             A(J,K)=TT(J,K,2)
             A(J+2,K)=0
             Z=MIN(Z,A(J,K))
           ENDIF
 800       CONTINUE
           DO 850 J=1,10
           DO 850 K=1,10
 850        A(J,K)=A(J,K)-Z
           CALL ZX3LP(A,IA,B,C,N,M1,M2,V,PSOL,DSOL,RW,IW,IER)
           DO 870 II=1,10
             STR(II)       =DSOL(II)/V
 870         STR(II+10)     =PSOL(II)/V
           GVA=Z+1/V
              IF (ISTAG.EQ.NST) THEN
           GG=GVA*REDF
             PRINT *, ' TRY TO BEAT TOTAL GAME VALUE OF ',GVA*REDF
               ENDIF
         CALL GGUBS(NL,1,BA)
          ST=0
          DO 875 NPL=IPL,IPL+9
           ST=ST+STR(NPL)
           IF (BA.LE.ST) GO TO 879
 875      CONTINUE
           ENDIF
```

```
879          III=ISTAG/3
              IF((III*3.EQ.ISTAG).OR.(NOP.EQ.2).OR.(ISTAGE.EQ.NST))
THEN
             PRINT *,'STRATEGY #',(IJ,IJ=1,10),'  BLUE   RED'
             PRINT *,'CAS         ',(SS(IJ,1),IJ=1,10),'    ',BCAS,'
',RCAS
             PRINT *,'ABA         ',(SS(IJ,2),IJ=1,10),'    ',BABA,'
',RABA
             PRINT *,'AD          ',(SS(IJ,3),IJ=1,10),'    ',BAD,'      '
,RAD
          ENDIF
             IF (NPL.GT.10)THEN
               PRINT *, 'RED  STRATEGY IS READY,PRINT YOUR STRATEGY'
        PRINT *
               READ *,J
             K=NPL-10
               PRINT *,'RED PLAY #',NPL-10
             ELSE
               PRINT *,'BLUE STRATEGY IS READY,PRINT YOUR STRATEGY #'
        PRINT *
               READ *,K
             J=NPL
               PRINT *,'BLUE PLAY #',NPL
             ENDIF
               RDF=REDF
         IF ((NOP.EQ.2).AND.(BLUF.NE.0.).AND.(REDF.NE.0.))THEN
           PRINT *,'BLUE PROB. :',(II,STR(II),II=1,10)
           PRINT *,'RED PROB.  :',(II-10,STR(II),II=11,20)
           ENDIF
            T=(SS(J,1)*BLUF*BCAS/3.-SS(K,1)*REDF*RCAS/3.)
         REDF= MAX(0.,REDF-MAX(0.,SS(J,2)*BI*BABA/3.-SS(K,3)*RAD/3.
) *RDF)
         BLUF= MAX(0.,BLUF-MAX(0.,SS(K,2)*RABA/3.-SS(J,3)*BI*BAD/3.
) *RDF)
             PRINT *,'  GAME VALUE = ',T, '  RED FORCE = ',REDF
             PRINT *,'       BLUE FORCE = ',BLUF
           TG=TG+T
             PRINT *, '  TOTAL GAME VALUE = ',TG
         PRINT*
1000      CONTINUE
          PRINT *,'  THE OPTIMAL GAME VALUE WAS = ',GG
          PRINT *,' END OF THE GAME '
             PRINT *
             PRINT *, ' IF YOU WANT TO PLAY ANOTHER GAME,PRINT 1  '
        PRINT *
             READ *, JJ
           IF (JJ .EQ.1) GO TO 2000
         END
*EOR
```

Appendix D

Computer Code for Linear Algorithm with M=3 (i.e., 10 Strategies)

```
   PROGRAME TAPE
    REAL TT(40,10,10),SS(10,3),STR(80),X(40),Y(40)   , B(12),C(10),
   :V,PSOL(10),DSOL(12),RW(178) ,YY(42) ,BI ,A(12,10),T
   REAL CU(40,3)
   INTEGER IA,N,M1,M2,IW(40),IER,NSTAGE,L,M ,I,J,K ,ISTAG
   DATA NSTAGE/15/,RABA/1/,BABA/1/,RAD/1/,BAD/1/,RCAS/1/,BCAS/1/
   :,NF/29/,BNKILD/.000001/,RNKILD/.000001/
   WRITE  (4,*) NSTAGE,NF
   DO 3000 IBLC=1,10
   PRINT *,'IBLC    ',IBLC
   IF (IBLC .EQ.1)THEN
   ELSEIF(IBLC.LT.5)THEN
   BCAS=IBLC/2.+.5
   ELSEIF (IBLC.LT.8) THEN
   BABA=(IBLC-2)/2.
   BCAS=1
   ELSE
   BABA=1
   BAD=(IBLC-5)/2.
   ENDIF
24  CONTINUE
   DATA SS/3,2,2,1,1,1,4*0.0,0,1,0,2,1,0,3,2,1,0,0,0.0,1,0,1,2,0,1,2,
   :3/M1/10/,M2/0/,N/10/,IA/12/,B/12*1/,C/10*1/
   DO 1000 ISTAG=1,NSTAGE
     DO 900 I=1,NF
       IF (I.LE.7) THEN
        BI=-.05+.05*I
        ELSEIF (I.LE.9) THEN
        BI=-.4+.1*I
        ELSEIF(I.LE.15)THEN
        BI=-1.75+.25*I
       ELSE
        BI=-5.5+.5*I
       ENDIF
       X(I)=BI
       Z=0.0
       DO 800 J=1,10
       DO 800 K=1,10
        IF (ISTAG.EQ.1) THEN
         TT(I,J,K)=SS(J,1)*BI*BCAS/3.-SS(K,1)*RCAS/3.
         A(J,K)=TT(I,J,K)
         A(J+2,K)=0
         Z=MIN(Z,A(J,K))
        ELSE
         RED=MAX(0.,(SS(J,2)*BI/3.-SS(K,3)*RAD/3.)*BABA)
         BLUE=MAX(0.,(SS(K,2)/3.-SS(J,3)*BI*BAD/3.)*RABA)
```

```
              T2=MAX(RNKILD,(1-RED))
              T=MAX(BI*BNKILD,(BI-BLUE))/MAX(RNKILD,(1-RED))
          IF(RED.GE.1)THEN
             PAY=MAX(0.,(BI-BLUE)*(ISTAG-1))
             T2=1
               ELSEIF((T.GE.X(NF-1)).AND.T.LT.X(NF))THEN
                 PAY=Y(NF-1)+(Y(NF)-Y(NF-1))*(T-X(NF-1))/(X(NF)-X(NF-1))
                ELSEIF(T.LT.X(2))THEN
                  PAY=Y(1)+(Y(2)-Y(1))*(T-X(1))/(X(2)-X(1))
                ELSEIF (T.LT.X(NF-1)) THEN
                  M=1
                  IC=42
                  CALL ICSCCU(X,Y,NF,CU,IC,IER)
                  CALL ICSEVU(X,Y,NF,CU,IC,T,PAY,M,IER)
                ELSE
               PAY=Y(NF)+(Y(NF)-Y(NF-1))*(T-X(NF))/.5
               ENDIF
 100           A(J,K)=TT(I,J,K)+PAY*T2
               A(J+2,K)=0
               Z=MIN(Z,A(J,K))
              ENDIF
 300        CONTINUE
            DO 850 J=1,10
            DO 850 K=1,10
 850         A(J,K)=A(J,K)-Z
            IF (I.EQ.1) THEN
              Z=-1-ISTAG
              V=1
              PSOL(1)=1
              DSOL(1)=1
              DO 860 I1=2,10
                PSOL(I1)=0
 860            DSOL(I1)=0
             ELSE
            CALL ZX3LP(A,IA,B,C,N,M1,M2,V,PSOL,DSOL,RW,IW,IER)
            ENDIF
            DO 870 II=1,10
              STR(II)       =DSOL(II)/V
 870          STR(II+10)     =PSOL(II)/V
             IF(ISTAG.GT.2)THEN
               ENDIF
            YY(I)=Z+1/V
 900       CONTINUE
```

```
              DO 950 II=1,NF
          Y(II)=YY(II)
950       CONTINUE
        IF((ISTAG.EQ.1).AND.(IBLC.EQ.1))THEN
        WRITE (4,*)(X(II),II=1,NF)
        ENDIF
        WRITE (4,*) (Y(II),II=1,NF)
1000     CONTINUE
3000     CONTINUE
         END
*EOR
```

```
      PROGRAME PRINT
      REAL FATEEN(40,20,15),X(42),Y(42),Z(42)
      REWIND 4
      READ(4,*) NSTAG,NF
      READ(4,*) (X(II),II=1,NF)
      DO 100 IBLC=1,10
      DO 100 ISTAG=1,NSTAG
      READ (4,*) (FATEEN(II,ISTAG,IBLC),II=1,NF)
100    CONTINUE
      DO 200 IBLC=1,10
      PRINT *
      PRINT *
      PRINT *
      PRINT *,' FR STG1  STG2  STG3  STG4  STG5  STG6  STG7  STG
STG9
     : STG10'
      DO 200 II=1,NF
        PRINT '(1X,F5.2,10F6.2)' ,  X(II),(FATEEN(II,ISTAG,IBLC)
ISTAG=1
     :,10)
200    CONTINUE
      END
      PROGRAM CURVES
      REAL FATEEN(40,20,15),X(42),Y(42),Z(42)
      REWIND 4
      READ(4,*) NSTAG,NF
      READ (4,*) NSTAG,NF
      READ(4,*) (X(II),II=1,NF)
      DO 100 IBLC=1,10
      DO 100 ISTAG=1,NSTAG
      READ (4,*) (FATEEN(II,ISTAG,IBLC),II=1,NF)
100    CONTINUE
      X(NF+1)=0.
      X(NF+2)=1.
      Y(NF+1)=-10.
      Y(NF+2)=10.
      CALL PLOTS(0.,0.,9)
      CALL PLOT (0.,0.,3)
      CALL PLOT (1.5,1.0,-3)
      CALL FACTOR(.7)
      DO 400 IB=1,10
      CALL AXIS(0.,1.,'FORCE RATIO',-11,9.,0.,X(NF+1),X(NF+2))
      CALL AXIS(0.,0.,'GAME VALUE',10,13.,90.,Y(NF+1),Y(NF+2))
      DO 300 ISTAG=1,NSTAG
      DO 200 I=1,NF
200       Y(I)=FATEEN(I,ISTAG,IB)
      CALL PLOT (0.,0.,3)
      CALL LINE (X,Y,NF,1,126)
300     CONTINUE
      CALL SYMBOL(2.,15.,.5,'CAS =',5)
      CALL SYMBOL(2.,14.,.5,'ABA =',5)
      CALL SYMBOL(2.,13.,.5,'AD  =',5)
      PRINT *,IBLC
      READ *,IIII
400    CONTINUE
      CALL PLOTE(11.,0.,3)
      END
```

Appendix E

Computer Code for OPTSA

```
      PROGRAM OPTSA
      REAL AAA(12,10),AA(12,10),A(12,10),SS(10,3),PSOL(10),
     :DSOL(10),RW(178),C(10),B(12)
      INTEGER IW(40)
      PRINT*,'BLUE,RED,BABA,RABA,BAD,RAD,BCAS,RCAS'
      READ *,BLUE,RED,BABA,RABA,BAD,RAD,BCAS,RCAS
      PRINT *,'BR,RR'
      READ *,BR,RR
      DATA SS/3,2,2,3*1,5*0,1,0,2,1,0,3,2,1,0,0,0,1,
     :0,1,2,0,1,2,3/,M1/10/,M2/0/,N/10/,IA/12/,B/12*1/,C/10*1/
      ZZZ=0
      DO 1000 IB=1,10
      DO 1000 IR=1,10
      BLU1=BLUE-MAX(0.,(SS(IR,2)*RED   -SS(IB,3)*BLUE*BAD)*RABA/3.)
      BLU1=MAX(BLU1,BR*BLUE)
      RED1=RED-MAX(0.,(SS(IB,2)*BLUE   -SS(IR,3)*BLUE*RAD)*BABA/3.)
      RED1=MAX(RED1,RR*RED)
*
*
*
*
      CAS1=SS(IB,1)*BLUE*BCAS/3.-SS(IR,1)*RED*RCAS/3.
      ZZ=0
      DO 100 JB=1,10
      DO 100 JR=1,10
      BLU2=BLU1-MAX(0.,(SS(JR,2)*RED1/3.-SS(JB,3)*BLU1*BAD/3.)*RABA)
      BLU2=MAX(BLU2,BR*BLU1)
      RED2=RED1-MAX(0.,(SS(JB,2)*BLU1/3.-SS(JR,1)*RED1*RAD/3.)*BABA)
      RED2=MAX(RED2,RR*RED1)
*
*
*
*
      CAS2=CAS1+SS(JB,1)*BLU1*BCAS/3.-SS(JR,1)*RED1*RCAS/3.
      Z=0
         DO 50 KB=1,10
         DO 50 KR=1,10
         CAS3=CAS2+SS(KB,1)*BLU2*BCAS/3.-SS(KR,1)*RED2*RCAS/3.
         Z=MIN(Z,CAS3)
      A(KB,KR)=CAS3
50       CONTINUE
         DO 70 KB=1,10
         DO 60 KR=1,10
*
60     A(KB,KR)=A(KB,KR)-Z
      A(11,KB)=0
70     A(12,KB)=0
      CALL ZX3LP(A,IA,B,C,N,M1,M2,V,PSOL,DSOL,RW,IW,IER)
      IF(V.EQ.0) THEN
      PRINT *,'V=0'
      ELSE
      AA(JB,JR)=Z+1/V
      ENDIF
90      ZZ=MIN(ZZ,AA(JB,JR))
100       CONTINUE
      DO 150 JB=1,10
      DO 150 JR=1,10
150        AA(JB,JR)=AA(JB,JR)-ZZ
      CALL ZX3LP(AA,IA,B,C,N,M1,M2,V,PSOL,DSOL,RW,IW,IER)
      AAA(IB,IR)=ZZ+1/V
900    ZZZ=MIN(ZZZ,AAA(IB,IR))
1000     CONTINUE
      DO 1100 IB=1,10
      DO 1100 IR=1,10
1100        AAA(IB,IR)=AAA(IB,IR)-ZZZ
      CALL ZX3LP(AAA,IA,B,C,N,M1,M2,V,PSOL,DSOL,RW,IW,IER)
      GV=ZZZ+1/V
      PRINT '(2X,10F7.5)',(PSOL(J)/V,J=1,10)
      PRINT '(2X,1F7.5)',(DSOL(J)/V,J=1,10)
      PRINT *
       PRINT *,'GAME VALUE = ',GV
      DO 1200 I=1,10
      PRINT '(2X,10F7.2)',(AAA(I,J)+ZZZ,J=1,10)
1200        CONTINUE
      END
```

Bibliography

1. Berkovitz, Leonard, and Melvin Dresher. Optimal Employment of Tactical Air in Theater Air Tasks-II, A GAME Theoretic Analysis. RM-2137. The Rand Corporation, March 1958.

2. Bracken, Jerom. Two Optimal Sortie Allocation Models, Volume I, Methodology and Results. Paper P-992. Institute for Defense Analyses, December 1973. AD 771-779.

3. Dresher, Melvin. Games of Strategy Theory and Applications. The Rand Corporation, 1961.

4. Foley, John M. A Two Person Simulated Tactical Air War Game. Air Force Institute of Technology, Wright-Patterson AFB, Thesis, March 1980.

5. Forrester, D. E. Industrial Dynamics. Cambridge: MIT Press, 1961.

6. Galiaro, R. J. Results of a Survey of Tactical Air Campaign Models. Arlington, Virginia: Kehon, Inc., November 1974.

7. Hillier and Lieberman. Introduction to Operations Research. New York: Holden-Day, Inc., 1980.

8. Hornbeck, Robert W. Numerical Methods. New York: Quantum Publishers, Inc., 1975.

9. Lansdowne, Zachary F. Development of an Algorithm to Solve Multi-stage Games. Chapter III: Computational Experience. Palo Alto, California: Control Analysis Corporation, May 1973.

10. Nosenzo, L. V. Prototype of a Theater-level Tactical Combat Model. Torrance, California: Lulejian and Associates, February 1974. AD 920 793.

11. Shannon, Robert E. Systems Simulation The Art and Science. Englewood Cliffs, New Jersey: Prentice Hall, Inc., 1975.

12. Taha, Hamdy A. _Operations Research_. New York: Macmillan Publishing Company, Inc., 1971.

13. Van Horn, Richard L. "Validation of Simulation Results," _Management Science_, _17_:247-257 (January 1971).

Vita

Mohamed Abdelrahman Fateen was born in Cairo, Egypt, on June 2, 1946. He graduated from Orman High School in 1963 and attended Cairo University from which he graduated in 1968 with a Bachelor of Science in Aeronautical Engineering. He entered the Egyptian Air Force in 1969 and attended the Egyptian Military Technical College from which he graduated in late 1969 with a Bachelor of Military Science. Upon graduation he was assigned as an instructor in Aircraft Technical Training Center, Cairo, Egypt. He was selected to study Operations Research in the Egyptian Military Technical College in 1978. He received his diploma in 1980. He entered the United States Air Force Institute of Technology's Graduate Program in Operations Research in June 1980.

He is married to Sohair Abdelhaleem Elhag. They presently have three children: Abdallah, Heba and Omer.

Permanent Address: #20 Street 84
Maady
Cairo, Egypt

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFIT/GOR/MA/81D-5 · 2. GOVT ACCESSION NO. AD-A115 536	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A COMPUTERIZED ALGORITHM FOR SOLVING MULTI-STAGE SIMULTANEOUS GAMES	5. TYPE OF REPORT & PERIOD COVERED Master's Thesis
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Mohamed Abdelrahman Fateen	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Air Force Institute of Technology (AFIT/EN) Wright-Patterson AFB, Ohio 45433	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS	12. REPORT DATE December 1981
	13. NUMBER OF PAGES 135
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)	15. SECURITY CLASS. (of this report) UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

15 APR 1982

Dean for Research and Professional Development
Air Force Institute of Technology (ATC)
Wright-Patterson AFB, OH 45433

[illegible]

18. SUPPLEMENTARY NOTES

APPROVED FOR PUBLIC RELEASE; IAW AFR 190-17

FRED~~RIC C. LYNCH, Major,~~ USAF
Director of Public Affairs

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

GAME THEORY	MATHEMATICAL MODELS
OPERATIONS RESEARCH	EFFECTIVENESS
AIR GAME	LINEAR PROGRAMMING
WAR GAME	MILITARY STRATEGY
DYNAMIC PROGRAMMING	COMPUTER PROGRAMMING

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The algorithm developed in this study finds the solution to multi-stage simultaneous games. A dynamic programming approach is used to solve the multi-stage game. The main idea of the solution is to build at each stage a matrix whose values are the payoffs obtained by playing each of the given strategies at this stage and the optimal strategies for the remaining stages. This payoff matrix is then solved using a linear programming algorithm.

A tactical air war problem was selected to illustrate the usefulness of the algorithm. An interactive GAME was developed for training and assisting the field commander. Finally, it was shown how the multi-stage algorithm can be used to compare two weapon systems by letting each side play its optimal strategy.